TUGAS AKHIR – TF 141581

RANCANG BANGUN ROBOT VISION UNTUK PENEMU OBJEK MENGGUNAKAN PENJEJAK WARNA PADA PERANGKAT RASPBERRYPI

ACHMADI
NRP. 2410 100 085

DOSEN PEMBIMBING :
Ir. Apriani Kusumawardhani, M.Sc.

JURUSAN TEKNIK FISIKA
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2015

FINAL PROJECT – TF 141581

VISION ROBOT DESIGN FOR OBJECT FINDING USING COLOR TRACKING ON RASPBERRYPI DEVICE

ACHMADI
NRP. 2410 100 085

SUPERVISOR :
Ir. Apriani Kusumawardhani, M.Sc.

Engineering Physic Departement
Faculty of Industrial Technology
Sepuluh Nopember Institute of Technology
Surabaya 2015

RANCANG BANGUN ROBOT VISION UNTUK PENEMU OBJEK MENGGUNAKAN PENJEJAK WARNA PADA PERANGKAT RASPBERRYPI

TUGAS AKHIR

Oleh :
ACHMADI
NRP : 2410 100 085

Surabaya, 23 Januari 2015
Mengetahui/ Menyetujui
Pembimbing

Ir. Apriani Kusumawardhani M. Sc
NIPN. 19530404 197612 2 001

Ketua jurusan
Teknik Fisika FTI-ITS

Dr. Ir. Totok Soehartanto. DEA
NIPN. 19650309 199002 1 001

RANCANG BANGUN ROBOT VISION UNTUK PENEMU OBJEK MENGGUNAKAN PENJEJAK WARNA PADA PERANGKAT RASPBERRYPI

TUGAS AKHIR
Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Teknik
pada
Bidang Studi Rekayasa Fotonika
Program Studi S-1 Jurusan Teknik Fisika
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember

Oleh
ACHMADI
NRP.2410 100 085

Disetujui oleh Tim Penguji Tugas Akhir :

1. Ir. Apriani Kusumawardhani M. Sc .....(Pembimbing)
2. Ir. Heru Setijono, M.Sc .....(Ketua Penguji)
3. Prof. Dr. Ir. Sekartedjo, M.Sc .....(Penguji I)
4. Dr. rer.nat. Ir. Aulia.M.T.N, M.Sc .....(Penguji II)
5. Detak Yan Pratama, ST, M.Sc .....(Penguji III)

SURABAYA
2015

RANCANG BANGUN ROBOT VISION UNTUK PENEMU OBJEK MENGGUNAKAN PENJEJAK WARNA PADA PERANGKAT RASPBERRYPI

Nama **: ACHMADI**
NRP **: 2410 100 085**
Jurusan **: TEKNIK FISIKA**
Dosen Pembimbing **: Ir. Apriani Kusumawardhani M.Sc**

ABSTRAK

Robot vision adalah robot yang mampu menggunakan kamera sebagai sumber informasi untuk diolah sesuai kebutuhan. Informasi yang ditangkap oleh kamera tersebut akan diolah oleh sebuah komponen pengolah informasi tersebut. Salah satu komponen pengolah informasi tersebut adalah RaspberryPi. Tujuan dari perancangan robot vision ini adalah untuk menjejak sebuah objek berdasarkan warnanya. Rancang bangun robot vision berupa komputer mini RasberryPi dilengkapi dengan kamera, motor controller, motor driver, motor dc.Kemudian dibangun perangkat lunak pengolah citra untuk menjejak warna dengan segmentasi berdasarkan nilai Hue 100-130, Saturation 87-182, dan Value 80-183. Kemudian dilakukan pengujian berupa run-test pada jarak terjauh, pemilihan bola diantara bola berwarna lain, uji medan pandang, kecepatan robot, dan uji pada tingkat pencahayaan yang berbeda. Berdasarkan eksperimen yang dilakukan dapat disimpulkan rancang bangun robot vision telah berhasil dilakukan dengan spesifikasi robot dapat menemukan objek uji berupa bola berwarna biru diantara objek lain dengan warna berbeda pada jarak maksimal 585 cm dan pencahayaan minimal 5 lux pada kecepatan putaran roda antara 126 hingga 445 rpm dengan medan pandang 58 derajat.

Kata Kunci— *Robot Vision, RaspberrPi, HSV, Penjejak Warna*

Halaman ini memang sengaja dikosongkan

VISION ROBOT DESIGN FOR OBJECT FINDING USING COLOR TRACKING ON RASPBERRYPI DEVICE

Name	**: ACHMADI**
NRP	**: 2410 100 085**
Departement	**: ENGINEERING PHYSIC**
Supervisor	**: Ir. Apriani Kusumawardhani M.Sc**

ABSTRACT

Vision Robot is a kind of robot that capable to utilize camera as information source for necessary processing. Information that collected by the camera will processed on a processing unit.One of processing unit is RaspberryPi.The purpose of this Vision Robot Design is tracking an object based on it's color.This design consist a RaspberryPi minicomputer equipped with a camera, motor controller, motor driver and DC geared motor.Then develops an image processing software for color tracking with segmentation based on Hue 100-130, Saturation 87-182, and Value 80-183.Then a test performed to get robot's spesification on farest distance, ball selecting on among other color, field of view, speed, and light intensity.According the experiments can concluded that Vision Robot was designed well with robot's spesifications can find a blue ball as testing object in 585 cm as maximal disctance and 5 lux as minimal lighting among other objects with different color with whell rotation speed between 126 rpm up to 445 rpm and field of view is 58 degree.

Kata Kunci— *Robot Vision, RaspberrPi, HSV, Color Tracking*

Halaman ini memang sengaja dikosongkan

KATA PENGANTAR

Alhamdulillah, puji syukur senantiasa terpanjatkan kepada Allah SWT yang Maha Segalanya. Atas rahmat, petunjuk dan kasih sayang-Nya, penulis mampu menyelesaikan Tugas Akhir dengan judul :

Rancang Bangun Robot Vision Untuk Penemu Objek Menggunakan Penjejak Warna Pada Perangkat Raspberrypi

Dalam proses menyelesaikan seluruh pengerjaan tugas akhir ini penulis mendapatkan banyak bantuan dan dukungan dari berbagai pihak. Penulis mengucapkan banyak terima kasih kepada:

1. Bapak Dr.Ir. Totok Soehartanto, DEA selaku Ketua Jurusan Teknik Fisika ITS.
2. Ibu Ir Apriani Kusumawardhani M Sc. selaku pembimbing yang telah memberi banyak ilmu, pengetahuan, wawasan dan bimbingan moral.
3. Bapak Prof. Dr. Ir. Sekartedjo, M.Sc selaku Kepala Laboratorium Rekayasa Fotonika yang telah memberi banyak ilmu, pengetahuan, wawasan dan bimbingan moral.
4. Ibu Dyah Sawitri ST, MT selaku dosen wali yang telah memberi banyak ilmu, pengetahuan, wawasan dan bimbingan moral
5. Bapak dan Ibu dosen Teknik Fisika yang telah memberi banyak ilmu, pengetahuan, wawasan dan bimbingan moral
6. Ibu, Bapak, dan adik yang telah memberikan banyak motivasi, doa, dan finansial yang sangat berharga.
7. Angkatan F-45, F-46, dan F47 yang saya banggakan atas motivasi, dukungan dan doanya.
8. Komunitas OpenSources Indonesia atas panduan teknis dan resources teknologi
9. Perkumpulan Pengembang RaspberryPi Indonesia atas panduan teknis dan resources teknologi.
10. Semua mahasiswa tugas akhir bidang Rekayasa Fotonika atas kebersamaan perjuangannya.

11. Semua mahasiswa warga Laboratorium Rekayasa Fotonika atas dukungannya.
12. Semua pihak yang tidak dapat penulis sebutkan satu per satu yang turut membantu dan memperlancar pengerjaan tugas ini. Terima Kasih yang sebesar-besarnya semoga Allah SWT membalasnya dengan pahala yang berlebih. Amin.

Penulis menyadari bahwa tugas akhir ini bukanlah suatu hasil yang sempurna, hanya harapan agar tugas ini menjadi referensi bagi rekan-rekan untuk menambah wawasan bagi pembaca dan dapat digunakan sebagai referensi pengerjaan tugas akhir selanjutnya. Semoga yang sederhana ini dapat menjadi motivasi untuk berkembang lebih hebat lagi.

Surabaya, Maret 2015

Penulis

DAFTAR ISI

DAFTAR TABEL

Halaman ini memang sengaja dikosongkan

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Robot vision adalah robot yang mampu menggunakan kamera sebagai sumber informasi untuk diolah sesuai kebutuhan (Browning,2013).Tujuan utama setiap perancangan robot tentu adalah untuk mengganti pekerjaan manusia.Kemampuan robot untuk melakukan pekerjaan yang berulang dan berbahaya telah menjadi kebutuhan di setiap lingkungan industri (Chao,2014).Untuk memenuhi kebutuhan tersebut telah dikembangkan beragam teknologi sensor dan actuator.Khusus untuk sensor, telah dikembangkan teknologi yang mirip dengan cara kerja pada indra manusia.Salah satu yang banyak dipakai adalah penggunaan kamera sebagai pengganti mata untuk robot (Browning,2013).

Penggunaan kamera sebagai sensor visual telah banyak di terapkan pada bidang robotika (Browning,2013).Sebagai contoh robot Asimo dengan harga $2.500.000 per unit menggunakan kamera CCD diproses dengan 20 CPU buatan Honda (Honda,2007).Kemudian robot Noa dengan harga $7.900 per unit menggunakan kamera SOC yang diproses dengan Intel Atom (Aldebaran,2012).Sedangkan robot Darwin-Op dengan harga $12.000 per unit menggunakan kamera CMOS diproses dengan Intel Atom (Robotis,2012).Berdasarkan informasi tersebut, maka harga tersebut masih tergolong mahal untuk aplikasi skala rumah dan kantor secara umum.

Selain kamera terdapat jenis-jenis sensor lain.Salah satu contohnya adalah jenis sensor proximity (kedekatan).Sensor proximity yang umum dipakai adalah sensor berbasis ultrasonik dan IR (Widodo, 2007).Kedua jenis sensor ini mampu mengukur jarak dekat antara sensor dengan sebuah objek/penghalang (obstacle).Karena sensor ini hanya mampu memberikan jarang maka sensor ini bagus untuk menghindar (avoiding), namun kurang untuk menemukan objek dengan kriteria tertentu.(Widodo,2007).

RaspberryPi merupakan salah satu jenis SBC (Single Board Computer) dengan spesifikasi processor BCM235 arsitektur arm1176 kategori armhf dengan clock 700MHz dan RAM 512.Tersedia

Operating System yang dapat dijalankan oleh RaspberryPi yaitu Raspbian yang berbasis Linux dan memiliki pustaka pengolahan citra OpenCV di lumbung perangkat lunaknya.RaspberryPi dan Operating System Raspbian merupakan proyek opensource yang bersifat gratis (Upton,2012).Perangkat RaspberryPi tersedia dengan harga cukup murah yaitu hanya $60 dan memiliki antar muka USB sehingga dapat berkomunikasi dengan kamera webcam.

Selain RaspberryPi, terdapat banyak SBC lain.Sebagai contoh adalah pcDuino, SBC yang bentuk fisiknya mirip dengan arduino.SBC pcDuino memiliki core ARM Cortex A8 dengan clock 1GHz dan DRAM 1GB.Melihat dari spesifikasi dapat terlihat bahwa pcDuino memiliki kemampuan lebih tinggi.Namun dari sisi harga, pcDuino relatif dua kali lipat dibandingkan dengan RaspberryPi. Selain itu, pcDuino membutuhkan arus 2000mA sehingga lebih boros tenaga dan lebih membutuhkan sistem pendinginan di CPU ketimbang RaspberryPi.Dengan pertimbangan kemampuan minimal yang dibutuhkan dengan harga dan konsumsi tenaga maka dipilihlah RaspberryPi.

Pustaka OpenCV (Open Computer Vision) merupakan pustaka pemrograman berbasis C/C++/Python yang berisi fungsi-fungsi untuk akuisisi dan pengolahan citra. Pustaka OpenCV juga merupakan proyek opensource yang bersifat gratis.Saat ini pustaka OpenCV telah diterapkan di banyak website dan aplikasi mobile untuk deteksi wajah dan penjejak warna (OpenCV,2014).

Penjejakan warna (Color Tracking) merupakan salah satu bentuk pengenalan objek yang cukup sederhana. Prinsip proses penjejakan warna adalah mensegmentasi gambar sesuai warna kemudian membaca perubahan koordinat warna yang tidak tersegmentasi relatif terhadap dimensi gambar.Prinsip penjejakan warna ini dapat digunakan untuk mengenali suatu objek dan membuang objek lain sesuai warna kemudian mengikuti perubahan pixel warna yang tidak tersegmentasi untuk mengetahui perubahan posisi objek yang dikenali.

Selain metode penjejakan warna, dikenal pula metode lain seperti penjejakan berbasis kontur.Dengan metode ini, gambar diolah berdasarkan pixel yang sama untuk mendeteksi garis tepi (Egde Detection).Setelah di dapat pola garis tepi maka kemudian di cari

pendekatan geometri garis tepi dengan persamaan matematika tertentu.Dibandingkan dengan penjejakan warna, penjejakan berbasis kontur tergolong lebih rumit karena memerlukan persamaan matematika untuk dapat mengenali pola sehingga mwembutuhkan banyak macam persamaan untuk bentuk yang berbeda-beda (Hermawati,2013).

1.2 Perumusan Masalah

Permasalahan dalam tugas akhir ini adalah bagaimana merancang dan membangun robot vision berbasis color tracking menggunakan perangkat RaspberryPi.

1.3 Batasan Masalah

Untuk mencegah meluasnya permasalahan maka ditentukanlah batasan masalah yang digunakan dalam tugas akhir ini yaitu:

- Warna objek yang menjadi bahan uji adalah warna yang ditentukan dalam program yang digunakan dalam robot.
- Penjejakan objek hanya berdasarkan warna, bukan berdasarkan bentuk, kontur, maupun dimensi.
- Algoritma robot hanya untuk mencari satu jenis warna, bukan dua atau lebih warna dalam waktu sama.
- Algoritma robot tidak dirancang untuk menjejak lebih dari satu objek.
- Robot bekerja pada tingkat pencahayaan yang relatif konstan.
- Perangkat Raspberry Pi dijalankan secara default tanpa overclock maupun modifikasi hardware lainnya.

1.4 Tujuan Tugas Akhir

Tujuan dari tugas akhir ini adalah merancang dan membangun robot vision berbasis color tracking pada perangkat RaspberryPi.

1.5 Manfaat Penelitian

Motivasi dilakukan penelitian ini adalah penelitian ini mempunyai prospek ke depan yang bermanfaat baik dalam segi aplikasi.Manfaat dari penelitian ini adalah hasil rancangan dasar sistem navigasi robot yang bersifat visual dengan biaya pengembangan yang relatif murah.Karena pada dasarnya rancangan ini hanya menggunakan web-kamera dan mini-komputer maka pengembangan rancangan ini tergolong mudah.Dengan rancangan dasar ini diharapakan di masa depan di setiap rumah tangga dapat membangun sendiri robot semacam ini untuk kebutuhan rumah tangga.Tidak menutup kemungkinan juga pengunaan di wilayah militer maupun perusahaan besar.

1.6 Sistematika Laporan

Laporan dari tugas akhir ini akan disusun dalam 5 bab sebagai berikut:

- **Bab I Pendahuluan**

Pendahuluan berisi tentang latar belakang masalah yang diambil, perumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika laporan.

- **Bab II Dasar Teori**

Dasar teori berisi tentang teori-teori yang terkait dengan tugas akhir ini.

- **Bab III Metode Penelitian**

Metode penelitian berisi tentang penjelasan mengenai desain robot dan desain pengujian.

- **Bab IV Analisa Data dan Pembahasan**

Analisa data dan pembahasan berisi mengenai hasil eksperimen yang dilakukan, analisis dari hasil yang didapatkan, dan pembahasan.

- **Bab V Penutup**

Penutup berisi tentang kesimpulan yang didapatkan dari hasil penelitian dan saran untuk melakukan penelitian selanjutnya.

BAB II
TEORI PENUNJANG

Pada bab ini akan dijelaskan mengenai beberapa teori penunjang dalam pembuatan robot berbasis kamera.

2.1 Robot Vision

Robot vision adalah robot yang mampu menggunakan kamera sebagai sumber informasi untuk diolah sesuai kebutuhan (Ude,2010).Tujuan utama setiap perancangan robot tentu adalah untuk mengganti pekerjaan manusia. Kemampuan robot untuk melakukan pekerjaan yang berulang dan berbahaya telah menjadi kebutuhan di setiap lingkungan industri (Chao,2014) . Untuk memenuhi kebutuhan tersebut telah dikembangkan beragam teknologi sensor dan actuator. Khusus untuk sensor, telah dikembangkan teknologi yang mirip dengan cara kerja pada indra manusia. Salah satu yang banyak dipakai adalah penggunaan kamera sebagai pengganti mata untuk robot (Browning,2013).

2.1.1 Robot

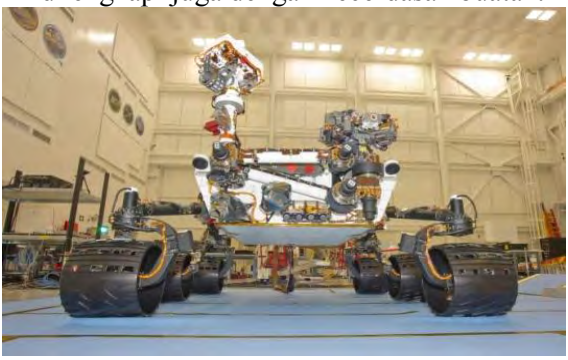
Robot adalah sistem otomasi tingkat lanjut yang menggunakan komputer sebagai bagian dari sistem kendalinya yang terintegrasi (Koren,2010). Secara umum, robot memiliki setidaknya 3 bagian penting yaitu sensor, sistem kendali, dan akuator. Sistem kendali dapat berupa sistem analog (berbasis Op-Amp) maupun sistem digital (berbasis chip tertanam atau komputer). Sensor robot sendiri juga dapat berupa informasi analog maupun digital. Sedangkan akuator robot sendiri beragam tergantung kebutuhan mulai dari derajat kebebasan satu hingga belasan. Gambar 2.1 berikut adalah blok diagram umum robot.

Gambar 2.1 Diagram blok robot.

2.1.2 Fungsi Robot

Fungsi Robot secara umum adalah untuk menggantikan tugas-tugas manusia. Tugas-tugas manusia tersebut digantikan karena sebab sebagai berikut (Kapila,1990):

- Pekerjaan tersebut berbahaya jika dilakukan oleh manusia. Sebagai contoh pekerjaan di medan berradiasi tinggi, luar angkasa, dan semacamnya. Gambar 2.2 berikut adalah foto unit Sojourner Rover sebelum dikirim ke planet Mars. Robot semacam ini memiliki kemampun untuk bekerja mandiri (autonomous) mengingat jarak yang jauh sehingga robot tidak mungkin dikontrol secara real-time dari bumi. Robot ini dilengkapi juga dengan kecerdasan buatan.



Gambar 2.2 Sojouner Rover

- Pekerjaan tersebut adalah pekerjaan yang berulang terus-menerus.Pekerjaan yang bersifat berulang-ulang apabila dikerjakan oleh manusia akan menjadi turun kualitas hasilnya seiring waktu.Pekerjaan tersebut juga akan menyebabkan manusia kelelahan baik secara fisik dan psikologi.Sebagai contoh pekerjaan industri seperti perakitan, pengelasan, dan semacamnya. Gambar 1.3 adalah foto tangan robot buatan ABB dalam sebuah jalur manufaktur mobil. Terlihat bahwa robot dapat melakukan pekerjaan berulang seperti pengelasan, pengecatan, pembautan, dan sebagainya. Sifatnya yang programmable membuat robot dapat membuat robot melakukan tugas yang

berbeda-beda.

Gambar 2.3 Tangan Robot untuk industri.

- Pekerjaan tersebut memang tidak harus dikerjakan oleh manusia.Seperti telah diketahui bahwa robot dan mesin pada umumnya hanya membutuhkan maintenance.Robot tidak memerlukan makan rutin, gaji tetap, tunjangan akhir bulan, asuransi dan semacamnya.Selain itu robot akan mematuhi perintah dan bekerja dengan kualitas yang cenderung konstan.Maka dengan menggunakan robot pada pekerjaan tersebut maka tingkat hasil pekerjaan akan meningkat dan biaya semakin mengecil.Sebagai contoh pekerjaan ini adalah pelayan kantor, pembersih lantai, dan semacamnya.

Gambar 2.4 Robot Asimo menggantikan tugas pelayan

2.1.3 Sensor Robot

Robot selayaknya manusia membutuhkan bagian-bagian untuk mengenali lingkungan atau mendapatkan informasi dari lingkungan. Berikut adalah pembagian sensor robot berdasarkan bentuk fisis informasi

Audio. Menggunakan informasi yang berupa gelombang merambat pada medium udara. Sebagai contoh adalah sensor ultrasonik untuk pengukur jarak dan microphone untuk memperoleh informasi berupa ucapan atau bunyi.

Mekanis. Menggunakan informasi yang berupa gaya-gaya mekanis. Sebagai contoh adalah taktil sensor untuk gaya benturan dan strain-gauge untuk infomasi tegangan.

Gelombang EM.Menggunakan informasi yang berupa gelombang merambat tidak perlu medium. Sebagai contoh adalah UVTron untuk memperoleh informasi UV atau photodioda untuk cahaya tampak.

Visual. Menggunakan informasi visual layaknya mata. Sensor yang digunakan berupa perangkat kamera.

2.1.4 Aktuator Robot

Aktuator robot adalah bagian robot yang dapat melakukan aksi. Bagian ini biasanya berupa sistem-sistem elektromekanik. Sistem elektromekanik adalah sistem mekanis yang dapat dikendalikan secara elektronik. Sebagai contoh adalah motor dc, motor brushless, motor stepper, motor servo, valve, dan relay.

2.1.5 Controller Robot

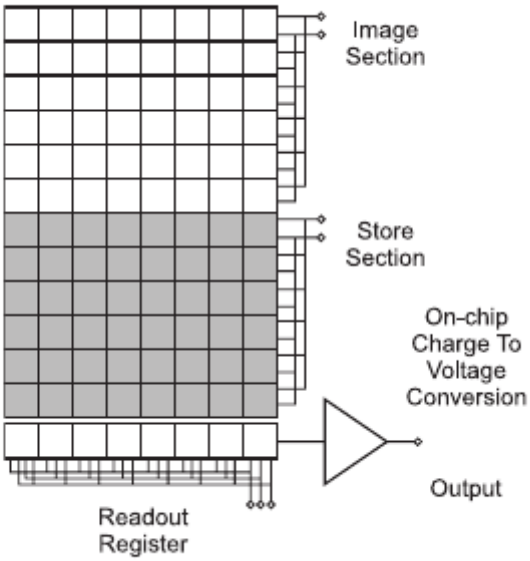
Controller robot adalah bagian yang penting karena disini dilakukan pengumpulan informasi dari sensor, pengolahan dan penentuan keputusan , serta penghasil sinyal untuk aktuator. Controller Robot dapat berupa analog maupun digital. Beberapa IC umum yang digunakan untuk Controller Robot adalah IC LM324, ATMega16, dan STM32F1. Selain IC, digunakan pula rangkaian seperti arduino, stmdiscovery, maupun raspberrypi sebagai Controller Robot.

2.2 Kamera

Kamera digital adalah instrumen perekam gambar (optis) secara elektronik.Beberapa jenis kamera digital yang saat ini banyak digunakan adalah jenis CCD, EMCCD, CMOS, dan ICCD.Dalam penelitian ini digunakan kamera berjenis CCD karena harga yang terjangkau dan populer. Kamera yang digunakan disini adalah Logitech C170.

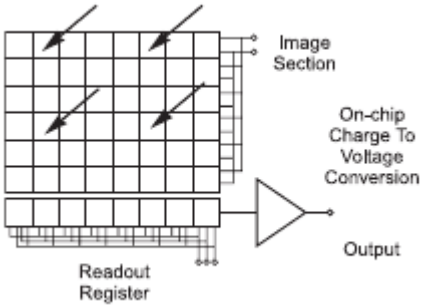
2.2.1 Sensor Kamera

Kamera CCD (Charge-Coupled Device) adalah jenis kamera dengan sensor dalam ukuran pixel berupa phototransistor yang terhubung penguat. Ketika paket cahaya atau foton membentur phototransistor tersebut, akan timbul muatan yang kemudian dikuatkan. Gambar 2.5 berikut adalah skematik setiap pixel dari sensor CCD.



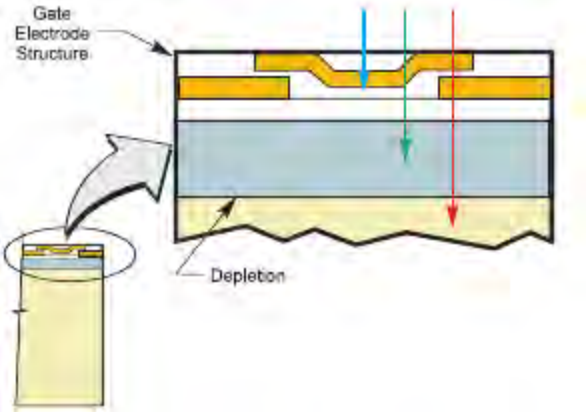
Gambar 2.5 Pixel sensor CCD

Ketika foton membentur sensor, setiap pixelnya menghasil elektron yang dibaca oleh Readout Register. Karena disusun dalam matrix maka dilakukan scanning dalam proses perekamannya.



Gambar 2.6 Pixel cahaya mengenai sensor CCD.

Setiap pixelnya, terdiri dari lapisan semikonduktor (silikon). Disetiap lapisan membaca R, G, dan B secara terpisah. Gambar 2.7 adalah skema sensor tiap pixel untuk memfilter setiap data warna dalam mode RGB.



Gambar 2.7 Filter RGB

2.2.2 Data Kamera Digital

Data kamera digital berupa data matrix 3 lapis dengan jumlah pixel sesuai resolusi kamera. Setiap elemen matrixnya berupa nilai 8 bit. Ketiga lapis matrix tersebut menunjukkan lapisan Red, Green, dan Blue sedangkan nilai setiap elemen menunjukkan intensitas setiap warna tersebut. Data matrix RGB ini kemudian dikirim melalui komunikasi serial baik berupa USB, I2C, SPI, maupun UART.

2.3 Mode Warna

Dalam pengolahan citra secara digital, dikenal beberapa jenis mode warna (color space). Mode warna ini adalah bentuk representasi matrix citra dalam model tertentu. Mode warna yang banyak digunakan adalah RGB, CMYK, HSL, dan HSV. Mode warna RGB merupakan mode warna dasar karena mode warna ini paling sederhana, sedangkan mode warna HSV sebagai mode yang dipilih karena mode warna ini merepresentasikan cara mata manusia melihat.

2.3.1 Mode warna RGB

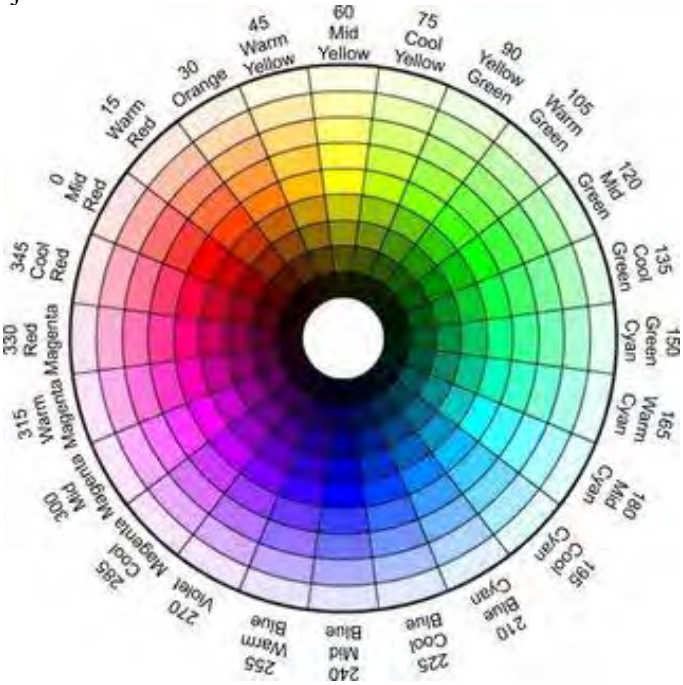
Mode warna RGB adalah mode warna yang dinyatakan dalam 3 variable yaitu Red, Green, Blue. Ketiga variabel tersebut memiliki rentang nilai yang sama yaitu 0 sampai 255 (bilangan 8bit). Nilai tersebut adalah nilai intensitas.Citra digital dalam format RGB memiliki 3 matrix yang saling bertumpuk masing-masing menunjukkan matrix R,G, dan B secara terpisah.

2.3.2 Mode warna HSV

Mode warna HSV adalah mode warna yang menyatakan warna dalam 3 variabel yaitu Hue, Saturation, dan Value.Nilai Hue merepresentasikan nilai jenis warna yang merupakan nilai kombinasi RGB dalam besaran sudut. Pada dasarnya nilai Hue dibagi dalam juring lingkaran sehingga jangkauan nilainya adalah 0-360, namun karena dalam bahasa pemrograman variabel pemrograman hanya 8bit (0-255) maka jangkauan juring di reduksi menjadi setengah

lingkaran (0-179).

Nilai Saturation merepresentasikan tingkat campuran suatu warna dengan warna putih dengan jangkauan nilai 0-255.Sedangkan nilai Value merepresentasikan tingkat campuran suatu warna dengan warna hitam dengan jangkauan nilai 0-255.Gambar 2.8 berikut menunjukkan mode warna HSV.



Gambar 2.8 Mode warna HSV

Pustaka OpenCV telah menyediakan fungsi untuk konversi RGB ke HSV. Konversi ini diperlukan karena hasil output kamera adalah matriks RGB sedangkan pengolahan selanjutnya menggunakan HSV.

Dalam konversi RGB ke HSV, nilai V didapat dari nilai terbesar di antara nilai R, G, dan B. Secara matematis dinyatakan dengan:

$$V = \max(R, G, B)$$

Untuk nilai S didapat dari rasio antara selisih V dikurangi

nilai minimal di antara R, G, dan B dibandingkan dengan nilai V itu sendiri.
Secara matematis untuk V > 0 dinyatakan dengan:

$$S = \frac{V - min(R,B,G)}{V}$$

Sedangkan untuk V=0 maka diambil S =0;

Untuk nilai Hue, maka konversi nilai RGB ke Hue dalam nilai bentuk sudut dinyatakan sebagai

$$H = \begin{cases} \frac{60(G-B)}{V-min(R,G,B)} \text{jikaV} = R \\ 120 + \frac{60(B-R)}{V-min(R,G,B)} \text{jikaV} = G \\ 240 + \frac{60(B-R)}{V-min(R,G,B)} \text{jikaV} = B \end{cases}$$

Konversi di atas menggunakan 360 derajat.Dalam pemrograman bilangan 8 bit hanya maximal 255 maka nilai 360 dibagi 2 sehingga kisaran nilai menjadi 0-179.

2.3.3 Citra Digital.

Citra Digital adalah media informasi dalam bentuk titik-titik warna tertentu.Titik-titk ini disebut pixel dan sekumpulan pixel menunjukkan informasi berupa citra atau gambar atau image. Citra digital disimpan dalam media bentuk nilai-nilai 8bit dalam sebuah berkas (file). Citra digital dihasilkan oleh perangkat kamera digital. Citra digital dapat berupa format TIFF, JPG, GIF, atau PNG

2.4 Color Tracking

Color Tracking adalah proses penentuan suatu koordinat objek dari sekumpulan pixel dengan warna tertentu dalam sebuah citra digital. Untuk dapat menentukan koordinat, maka diperlukan pemilihan dan pemisahan pixel-pixel dengan krteria tertentu. Proses ini disebut segmentasi.

Segmentasi citra adalah proses mengumpulkan pixel-pixel karena kesamaan properti.Kesamaan properti dapat berupa nilai intensitas, bentuk tekstur, garis tepi, dll.Ada banyak teknik untuk segmentasi citra dengan masing-masing memiliki kompleksitas, kemampuan, dan penggunaannya.

Proses segmentasi merupakan proses yang penting dalam pengolahan citra.Segmentasi digunakan untuk memfilter objek yang tidak dibutuhkan. Filter dapat berupa persamaan matematis,logic maupun maupun matrix kernel .

Proses segmentasi yang digunakan disini menggunakan Tresholding dimana setiap nilai pixel dibandingkan dengan nilai ambang.Jika masuk diantara ambang atas dan bawah, maka di ambil nilai putih (skala abu-abu), sedangkan jika tidak diambil nilai hitam (skala abu-abu).Secara matematis dinyatakan:

$$P(i,j) = \begin{cases} 255, jika Hmin < H < Hmax \wedge \\ Smin < S < Smax \wedge Vmin < V < Vmax \\ 0, jika selainnya \end{cases}$$

2.5 RaspberrPi

Raspberry Pi, sering juga disingkat dengan nama Raspi, adalah komputer papan tunggal (Single Board Circuit /SBC)yang memiliki ukuran sebesar kartu kredit. Raspberry Pi bisa digunakan untuk berbagai keperluan, seperti spreadsheet, game, bahkan bisa digunakan sebagai media player karena kemampuannya dalam memutar video high definition. Raspberry Pi dikembangkan oleh yayasan nirlaba, Rasberry Pi Foundation yang digawangi sejumlah developer dan ahli komputer dari Universitas Cambridge, Inggris.

2.5.1 Raspbian

Raspbian adalah Operating System berbasis Debian yang dijalankan di platform RaspberryPi.Raspbian menggunakan kernel Linux untuk arsitektur armhf.Melalui Raspbian ini nantinya menjalankan OpenCV untuk pengolahan citra dan komunikasi serial untuk mengontrol robot.Lumbung Software (Repository) dari Raspbian telah menyediakan binary siap pakai sehingga memudahkan instalasi.

2.5.2 Debian Software Manajer

Raspbian adalah salah satu OS yang merupakan dari turunan Debian. Setiap OS Debian dan turunannya memiliki metode manajemen software yang serupa. Manajemen software proses

instalasi atau uninstalasi software. Juga termasuk pengunduhan software dari lumbung software yang tersedia di internet. Untuk mempermudah instalasi, sebuah software seringkali di pecah menjadi bagian-bagian yang disebut paket software. File paket debain memiliki extensi *.deb.Berikut adalah nama program dalam manajemen software Debian:

- dpkg. Program ini berfungsi untuk instalasi atau uninsta suatu paket dan mendaftar setiap file yang berasal dari paket tersebut.
- apt-cache. Program ini berfungsi untuk mengtahui paket serta versinya yang tersedia di lumbung software.
- apt-get. Program ini berfungsi untuk menyelesaikan dependensi (ketergantungan) antar paket dan kemudian mendownloadnya dari lumbung software serta Kemudian memanggil program dpkg untuk instalasi.

2.6 OpenCV

OpenCV (Open Computer Vision) adalah pustaka pemrograman untuk pengolahan citra yang bersifat opensource.OpenCV mendukung bahasa pemrograman C, C++, dan Python.OpenCV dapat digunakan sebagai pengganti Matlab untuk proses pengolahan citra yang mengharuskan program ditulis dalam C atau C++.

2.6.1 Image Processing

OpenCV menyediakan fungsi-fungsi pengolahan citra yang lengkap mulai dari perhitungan histogram hingga algoritma Haar-Cascade untuk deteksi wajah. Fitur OpenCV yang digunakan di robot ini adalah:

- VideoCapture untuk memperoleh gambar dari kamera .
- Mat untuk menangani variabel matrix ukuran besar yang berisi data citra.
- CreateTrackbar untuk proses pengaturan nilai suatu variable dalam bentuk penggeser (slider).
- imshow untuk proses penampilan gambar.

- inRange untuk proses threshold nilai matrix.
- cvtColor untuk proses konversi warna.

2.6.2 QtextSerialPort

QextSerialPort adalah pustaka untuk komunikasi serial TTL yang dapat di kombinasikan dengan OpenCV untuk dapat memberi perintah kepada komponen selanjutnya setelah pengolahan citra selesai. QextSerialPort menyediakan fungsi write() untuk dapat mengirim perintah dalam data string.

2.7 Komponen Elektronik

Komponen elektronik yang dijelaskan disini adalah komponen pendukung yang digunakan robot selain komponen pada RaspberrPi dan kamera.

2.7.1 STM32F103RBT6

STM32F103RBT6 adalah chip produksi ST Microelectronics yang merupakan chip mikrokontroller 32bit. Chip ini memiliki arsitektur ARM (Advanced RISC Machine) Cortex-M3. Chip ini memiliki CPU dengan clock hingga 72MHz dengan memory SRAM 16kB dan memory flash 256kB. Fitur chip ini yang mendukung aplikasi sistem tertanam adalah ADC, GPT, PWM, dan ICU. Chip ini juga memiliki fitur komunikasi serial berupa I2C, SPI, USB, dan USART.

2.7.2 ChibiOS/RT

ChibiOS/RT adalah salah satu pustaka OS/RT (Operating System Real Time) yang dapat digunakan dalam pengembangan firmware dari chip STM32F103RBT6. Didalam pustaka ini telah terdapat fitur untuk melakukan proses multithread dan hardware abstraction.

2.7.3 FT232RL

FT232RL adalah chip antar muka serial-USB.Chip ini mengkonversi data serial ke data USB dan sebaliknya. Dengan penggunaan chip ini maka komunikasi antara komputer dan sistem

tertanam dapat dilakukan melalui port USB pada komputer dan jalur UART pada sistem tertanam.

2.7.4 Motor DC

Motor DC adalah jenis motor dengan tenaga listik berupa arus searah (Direct Current / DC). Motor ini adalah jenis motor listrik paling sederhana. Kendali motor ini adalah berupa polaritas untuk mengatur arah, sedangkan kecepatannya dapat dengan mengatur arus atau pulsa tegangan.

2.7.5 PC817

PC817 adalah chip optoswitch yang terdiri dari led dan phototransistor dalam 1 chip. PC817 biasa digunakan untuk rangkaian kontrol dengan tegangan antara kontroller dan driver terpisah.

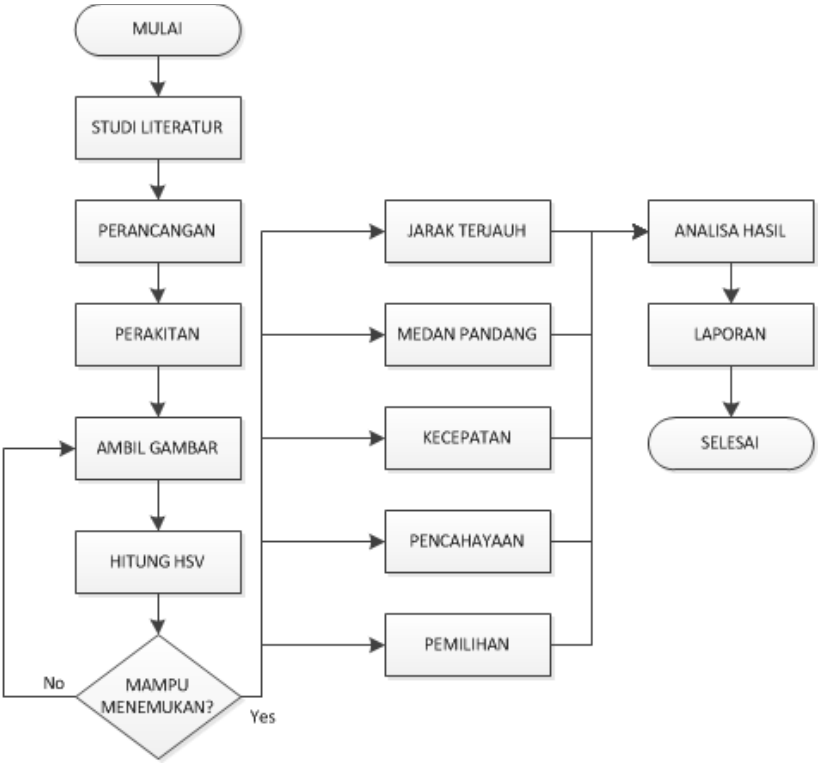
2.7.6 FET

Field Effect Transistor adalah jenis transistor dengan kendali berupa tegangan pada kaki Gate. Transistor jenis ini sering digunakan dalam rangkaian kendali arus cukup besar seperti kendali motor DC karena karakter dari FET yang mampu menangani arus hingga 5A. Selain itu sinyal kendali yang berupa tegangan mempermudah perancangan sistem kendali berbasis mikrokontroller karena mikrokontroller mengeluarkan sinyal berupa tegangan.

Halaman ini memang sengaja dikosongkan

BAB III
METODOLOGI PENELITIAN

3.1 Diagram Alir



Gambar 3.1 Diagram alir kerja

Penelitian ini dilakukan dengan beberapa tahapan hingga tujuan dapat tercapai. Pengerjaan tugas akhir ini meliputi studi literatur, perancangan dan pembangunan robot vision, pengujian robot vision, analisa data dan penyusunan laporan. Tahapan pengerjaan tugas akhir ini dimulai dengan studi literatur. Studi literatur ini bertujuan untuk mengetahui dasar-dasar perancangan

robot yang telah dibuat pengembang lain.Tahapan tersebut tergambar dalam diagram alir pada gambar 3.1 di atas.

Disini desain robot yang dipilih adalah desain robot beroda karena desain terebut adalah desain yang paling mudah dan paling dasar. Desain tersebut kemudian akan digunakan ketika perancangan. Setelah desain selesai dibuat, maka robot akan dirakit menjadi 1 unit. Setelah selesain perakitan, maka dilakukan proses penentuan nilai HSV sehingga robot dapat menemukan robot. Jika ada kegagalan dalam pencarian objek, maka dilakukan perbaikan. Kemudian unit tersebut akan di uji untuk mendapatkan spesifikasi dari robot itu sendiri mencakup jarak terjauh, jarak kiri-kanan,kecepatan, dan tingkat pencahayaan. Setelah didapatkan data tersebut, maka akan dilakukan analisa. Terakhir dilakukan penulisan laporan.

3.2 Perancangan Robot

Perancangan robot dimulai dari mendesain sistem robot yang terbagi menjadi bagian yaitu:

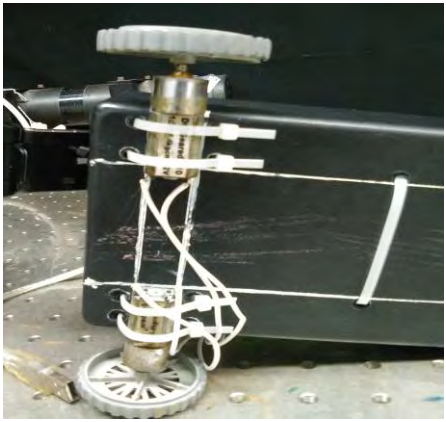
- Aktuator
- Driver motor
- Controller motor
- Sistem tenaga
- Algoritma perangkat lunak

Sedangkan untuk perangkat kamera dan RaspberryPi tidak perlu lagi dilakukan perancangan apa pun karena akan digunakan secara tanpa ada modifikasi.

3.2.1 Aktuator

Aktuator yang digunakan disini adalah motor DC dengan internal gear. Kemudian di as motor tersebut dipasang roda dengan ban karet. Berdasarkan label yang tertera di bodi motor diketahui bahwa motor DC tersebut memiliki torsi maximal 15 kg.cm dengan rpm maximal 510. Gambar 3.2 berikut menunjukkan motor dc yang telah terpasang di kotak robot. Motor DC ini hanya memiliki 2 jalur masukkan berupa tegangan. Motor ini tidak memiliki feedback untuk

kounter.



Gambar 3.2 Motor DC pada kotak

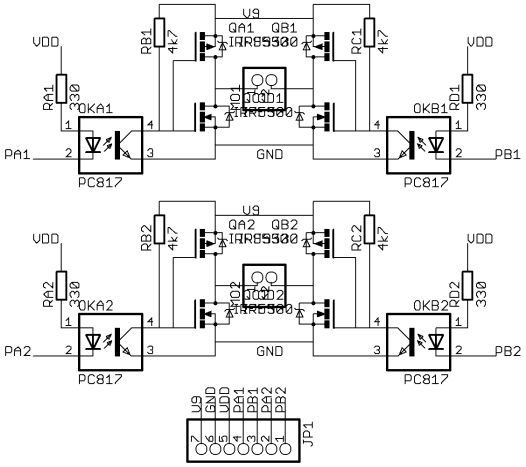
Alasan memilih tipe motor ini adalah:

- Ringan dan kecil.Dimensi motor yang hanya 1.5 x 5 cm dengan bobot 50 gr cocok untuk penggunaan robot ukuran kecil
- Internal Gear. Ratio torsi-rpm tidak memerlukan instalasi roda gigi karena telah tersedia.
- Daya minimal 12 volt 1 ampere. Dengan kebutuhan daya ini maka motor ini cukup untuk tenaga baterai LiPo ukuran kecil
- Murah (Rp. 90.000 per biji).

3.2.2 Driver Motor

Driver motor adalah bagian yang bertugas mengatur putaran motor baik arah maupun kecepatan rotasi. Komponen utama dari driver motor adalah IRF9530 dan IRF530 yang bertugas untuk switching. Sedangkan sebagai penyambung sinyal antara sistem digital dan driver adalah IC optoswitch PC817. Gambar 3.3 berikut adalah skema dari driver motor yang digunakan. Komponen berlabel

OKA1,OKA2, OKB1, dan OKB2 adalah optoswitch PC817 yang mendapat sinyal dari controller motor pada wiring PA1, PA2, PB1, dan PB2. Komponen berlabel QA1, QB1, QA2, dan QB2 adalah IRF9530 sedangkan QC1,QD1, QC2 dan QD2 adalah IRF530. Pin V9 dan GND terhubung ke baterai LiPo. Pin untuk sambungan motor adalah MO1 dan MO2.



Gambar 3.3. Skema driver motor

Kemampuan driver ini sesuai dari datasheet yang dimiliki oleh PC817, IRF9530, adalah IRF530 sebagai berikut:

Tabel 3.1 Spesifikasi komponen utama driver motor.

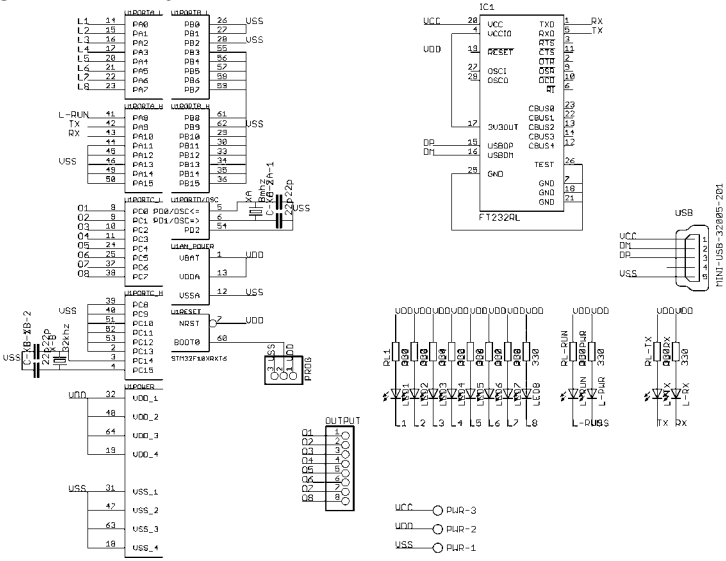
Aspek	IRF9530	IRF530	PC817
Tipe	P-type	N-type	NPN
Rating	100v 12A	100v 12A	12v 50mA
Sinyal	12v Act. Low	12v Act. High	TTL

Desain driver motor ini mengacu kepada desain H-brigde

tipe lurus sehingga membutukan 2 jenis switch, yaitu Active-Low (tipe P) dan Active-High (tipe N).

3.2.3 Controller Motor

Controller Motor adalah bagian yang bertugas memberi sinyal kepada driver motor untuk mengatur motor. Sinyal yang diberikan kepada driver berupa sinyal logic TTL dalam bentuk duty-cycle PWM (Pulse Width Modulation). Controller motor bekerja dengan menerima sinyal dari RaspberryPi berupa data serial standar TTL-UART. Karena RaspberryPi menyediakan jalur USB, maka diperlukan konverter USB ke serial TTL-UART. Komponen utama dari controller motor adalah mikrokontroller STM32F103RBT6 dan chip FT232RL sebagai konverter USB-serial TTL-UART serta rangkaian LED sebagai indikator. Gambar 3.4 berikut adalah skema controller motor yang digunakan. Desain skema ini merupakan contoh skema minimum sistem yang sudah umum untuk STM32 dengan tambahan jalur data serial-USB.



Gambar 3.4 Skema Controller Motor.

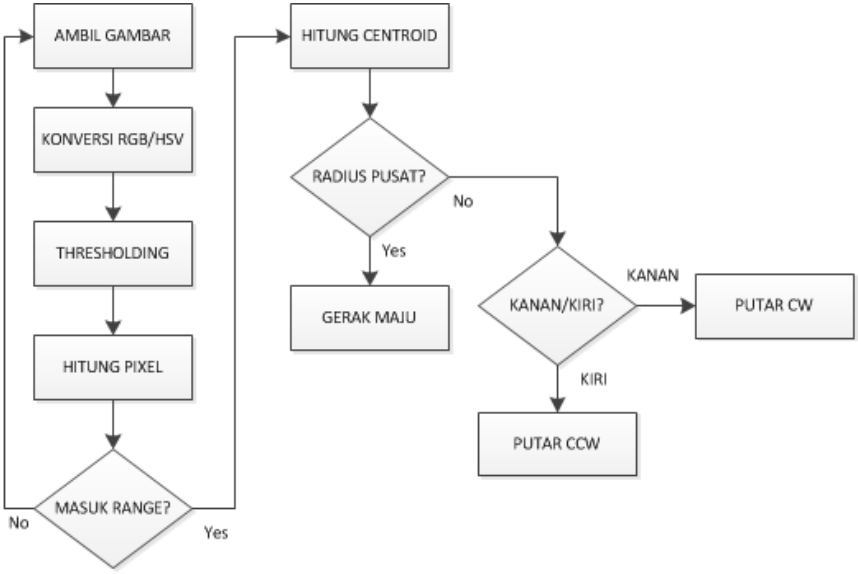
3.2.4 Sistem Tenaga

Sistem tenaga yang digunakan disini dibagi menjadi 2 kelompok:

- Arus besar sebagai tenaga penggerak motor DC. Sumbernya adalah baterai LiPOFe Rechargeable 11,1 v 1000mAh 25C merek Gens
- Arus lemah sebagai tenaga untuk sistem digital (Kamera, RaspberriPi, Controller Motor). Sumbernya adalah PowerBank 4400mAh 1A merek Advance seharga Rp.

3.2.5 Algoritma Perangkat Lunak

Algortima yang akan ditanamkan dalam perangkat RaspberriPi yang berfungsi sebagai pusat pengolah. Gambar 3.5 berikut adalah diagram algortima perangkat lunak yang akan digunakan.



Gambar 3.5 Diagram perangkat lunak.

Proses ambil gambar hingga penentuan pergerakan dilakukan oleh software berbasis C++ dengan pustaka OpenCV, sedangkan untuk perintah bergerak antara perangkat RaspberriPi dengan Controller motor menggunakan C++ dengan pustaka QtExtSerialPort..

3.3 Perakitan Robot

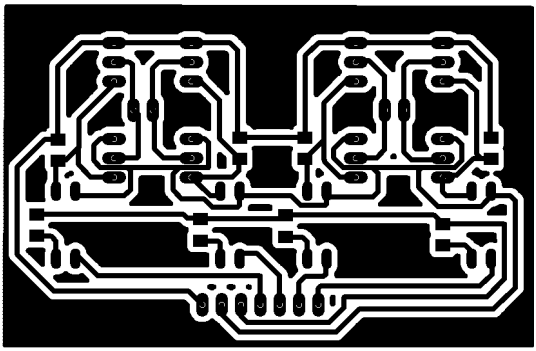
Perakitan robot adalah proses merealisasi semua desain setiap bagian kemudian merakitnya menjadi satu. Perakitan robot meliputi:

- Perakitan Driver motor
- Perakitan Controller motor
- Instalasi Sistem tenaga
- Instalasi OS pada perangkat RaspberryPi
- Instalasi Pustaka OpenCV dan QextSerialPort
- Kompilasi software
- Perakitan semua bagian dalam satu kotak.

3.3.1 Perakitan Driver motor

Perakitan driver motor dimulai dengan mencetak papan circuit yang akan digunakan. Proses pencetakan papan circuit diserahkan kepada PT. Maxtron Persada Indonesia. Gambar 3.6 adalah layout papan circuit yang dicetak. Dari bentuk layout tersebut dibuat jalur tembaga pada papan sirkuit yang terbuat dari fiber.

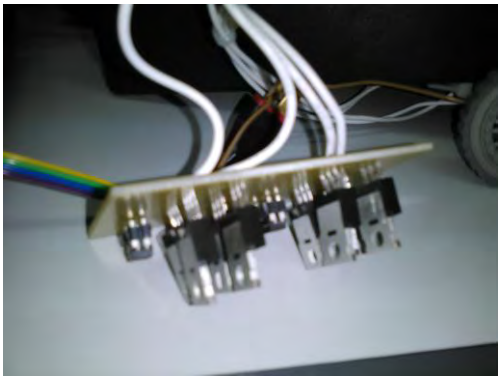
Proses pengerjaan memakan waktu 1 minggu dikarenakan terdapat komponen resistor paket SMD (Surface Mounted Device) tipe 1206. Setelah papan circuit selesai, diperlukan proses penenbalan dengan timah mengingat tebal lapisan tembaga hanya 0,05 mm. Lapisan tembaga ini sangat tipis sehingga dapat terbakar akibat dilewati arus dari baterai ke motor DC yang berkisar 1A hingga 2A apabila tidak dilapisi timah sebagai tambahan ketebalan dan pembuang panas.



Gambar 3.6 Layout sirkuit driver motor

Dalam proses wiringnya, setiap motor akan di wiring dengan masing-masing 2 pin header di tengah setiap H-brigde, sedangkan sinyal dari terhubung dengan 4 pin dari kiri header . Untuk jalur tenaga dari batere LiPo terhubung dengan 2 pin dari kanan. Gambar 3.7 berikut adalah gambar driver motor yang telah di wiring.

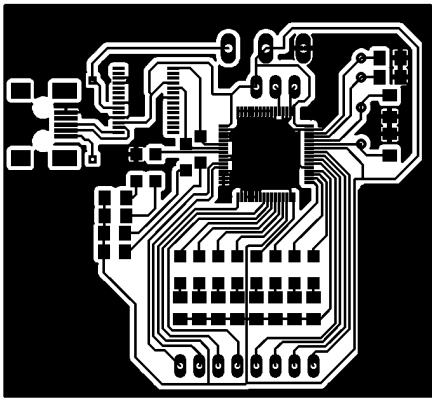
Digunakan 2 jenis kabel yaitu kabel pelangi sebagai penghantar tegangan sinyal antara kontroler dan driver, sedangkan jenis kabel kedua adalah NYAF 2mm sebagai penghantar tegangan daya antara baterai dan motor DC.



Gambar 3.7 Sirkuit driver motor yang telah terwiring

3.3.2 Perakitan Controller motor

Perakitan driver motor dimulai dengan mencetak papan circuit yang akan digunakan. Proses pencetakan papan circuit diserahkan kepada PT. Maxtron Persada Indonesia. Gambar 3.8 adalah layout papan circuit yang dicetak.



Gambar 3.8 Layout sirkuit controller motor

Dalam proses wiringnya, controller motor terhubung ke perangkat RaspberriPi melalui jalur USB, sedangkan dengan driver motor terhubung oleh pin 4 dari kiri header. Gambar 3.9 adalah gambar hasil rakitan. Proses pencetakan akan memakan waktu 1 minggu. Setelah papan circuit selesai tidak perlu dilakukan penebalan dengan timah karena arus yang mengalir dalam sistem ini tidak sampai 100mA. Namun perlu dilakukan masking mengingat jarak tiap pad solder kurang 1mm untuk mencegah adanya short yang dapat membuat sistem gagal.

Setelah sistem ini terakit maka dilakukan proses pemrograman. Dalam sistem ini diterapkan multithreading dimana proses komunikasi dengan raspberryPi, proses timer untuk sinyal PWM, dan proses utama CPU masing-masing berjalan dalam lingkup SRAM terpisah dan tersendiri dengan komunikasi antar thread melalui variabel global.

Gambar 3.9 Sirkuit controller motor

Perintah yang berasal dari perangkat RaspberriPi akan sampai ke controller motor melalui USB untuk kemudian diterjemahkan menjadi format serial. Controller motor telah diatur untuk menghasilkan sinyal ke driver motor berupa modulasi lebar pulsa (Pulse Width Modulation atau PWM) dengan besar duty-cycle yang konstan.

3.3.3 Sistem Tenaga

Dalam sistem tenaga, digunakan 2 baterai tanpa ada perlu wiring khusus. Konektor baterai LiPo langsung terhubung dengan driver motor dengan konektor din, sedangkan PowerBank terhubung langsung dengan perangkat RaspberriPi dengan kabel USB. Gambar 3.10 berikut adalah gambar untuk kedua baterai.

Baterai LiPo merupakan baterai rechargeable. Untuk recharge perlu diperhatikan bahwa jenis baterai ini dianjurkan untuk recharge dalam mode Balance dimana setiap selnya naik secara perlahan bersama-sama. Apabila ada sel yang terlalu rendah tegangannya sementara sel lain sudah tinggi, maka akan terjadi arus internal yang dapat membuat baterai rusak bahkan meletus.

Untuk baterai PowerBank telah memiliki sirkuit pengatur internal sehingga tidak perlu memperhatikan bagaimana proses recharge pada baterai ini.

Gambar 3.10 Baterai LiPo dan Power Bank

3.3.4 Instalasi OS

Perangkat RaspberryPi membutuhkan operating system untuk dapat bekerja. Disini dipilihlah Operating System Raspbian. OS Raspbian merupakan turunan dari OS Debian yang berbasis Linux. Perbedaan dengan Debian adalah Raspbian memiliki dikhususkan untuk RaspberryPi sedangkan Debian dapat digunakan di komputer secara umum. Software yang tersedia untuk Raspbian cukup banyak, termasuk juga OpenCV dan QtExtSerialPort

Untuk menginstal Raspbian maka perlu di download dahulu OS tersebut di http://www.raspbian.org. Setelah mendownload, maka file hasil download yang berformat *.zip hingga di dapat file yang berformat *.img. Kemudian melalui komputer lain file *.img tersebut dapat ditulis ke memory card dan dapat dijalankan. Perintah di komputer untuk menulis file format *.img ke memory card adalah:

ddrescue raspbian.img /dev/mmcblk0

Selanjutnya tunggu hingga selesai dan memory card akan berisi OS Raspbian. Program ddrescue akan menulis dan membentuk partisi sesuai konten image. Partisi yang terbentuk adalah dua partisi yaitu 50 MB (FAT32) untuk booting dan sisanya (ext4) untuk root

3.3.5 Instalasi Pustaka OpenCV dan QtExtSerialPort

Untuk dapat menginstal pustaka OpenCV dan QextSerialPort maka perlu dilakukan update pada perangkat RaspberriPi melalui internet. Setelah terhubung internet maka perintah berikut akan mengupdate RaspberryPi:

sudo apt-get update

Setelah selesai mengupdate, maka dilakukan instalasi beberapa paket melalui perintah berikut:

sudo apt-get install geany qtcreator libqt4-dev libopencv-dev

Selanjutnya setelah semua paket tersebut selesai di instal maka RaspberriPi dapat menjalankan pustaka OpenCV dan QextSerialPort.

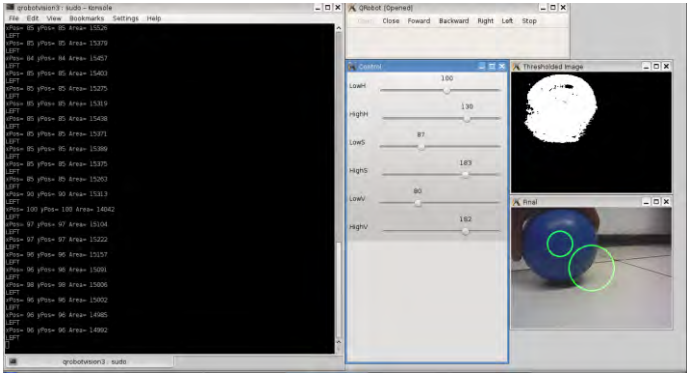
Pustaka OpenCV dan QextSerialPort ini akan dipanggil sebagai fungsi-fungsi dalam kode sumber C++ yang digunakan dalam pemrosesan gambar dan penentuan gerak robot.

3.3.6 Kompilasi Software

Karena disini menggunakan kode sumber C++ maka agar dapat digunakan perlu dilakukan kompilasi. Penulisan kode sumber dapat dilakukan di komputer umum, namum kompilasi nya dilakukan diperangkat RaspberriPi. Untuk mengkompilasinya, masukkan perintah-perintah berikut:

qmake
make
sudo cp -v ./qrobot /etc/xdg/autostart

Software hasil kompilasi jika dijalankan akan memiliki tampilan seperti berikut:



Gambar 3.11 Perangkat lunak untuk pemrosesan gambar.

3.3.7 Perakitan Total

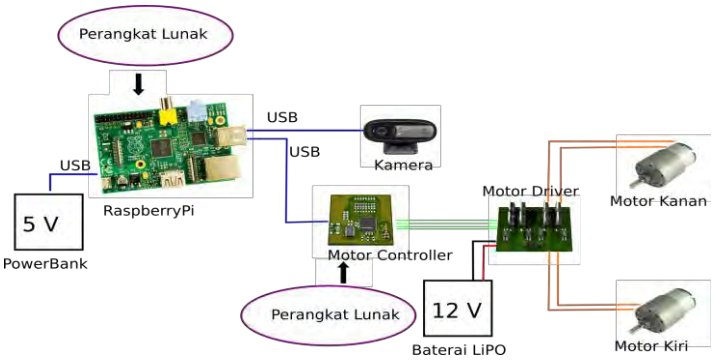
Setelah semua bagian dirakit dan software telah dapat dijalankan maka seluruh bagian dirakit menjadi 1 unit. Gambar 3.12 berikut adalah gambar robot yang telah terwiring namun masih belum terpasang dengan terintergrasi.

Gambar 3.12 Bagian-bagian robot.

Sedangkan gambar 3.13 berikut adalah tampilan robot secara total.

Gambar 3.13 Robot yang telah dirakit.

Dan gambar 3.14 berikut adalah skema hubungan antar komponen robot secara total.



Gambar 3.14. Skema robot secara total.

3.4 Penentuan jangkauan nilai HSV.

Untuk mendapatkan jangkauan nilai HSV, dilakukan pengambilan gambar objek dari jarak terdekat hingga jarak terjauh. Objek yang dipilih disini adalah bola dengan diameter 97,28 mm berwarna biru. Jarak terdekat adalah 0 cm dari depan kamera dan terjauhnya adalah 645 cm dengan kenaikan setiap 15 cm. Disetiap

jarak kemudian diambil gambar sebanyak 10 gambar, sehingga total ada 440 gambar.

Setiap gambar yang diperoleh kemudian dilakukan penentuan nilai jangkauan SV secara manual untuk mendapatkan pixel hasil threshold sebanyak mungkin, sedangkan nilai H sudah standar dari OpenCV untuk warna biru adalah 100-130. Data gambar sebanyak 440 data disajikan di lampiran.

Dengan mengambil rata-rata didapatkan Smin = 87, Smax = 183, Vmin = 80, dan Vmax = 182. Setelah didapat jangkauan nilai HSV maka dapat ditentukan bahwa nilai pixel terkecil adalah 31 sehingga jarak terjauh dapat ditetapkan pada 585 cm.

3. 5 Pengujian.

Untuk mendapatkan spesifikasi dari robot maka dilakukan pengujian meliputi:

- Run-test lurus pada jarak terjauh
- Medan Pandang (Field Of View)
- Kecepatan
- Pengaruh perubahan cahaya
- Pemilihan bola

Objek pengujian adalah bola dengan diameter 97,28 cm berwarna biru seperti pada gambar 3.15

Gambar 3.15 Bola warna biru untuk pengujian

3.4.1 Run-test lurus pada jarak terjauh

Pengujian ini di tujukan untuk menguji spesifikasi robot untuk menemukan objek lurus ke depan pada jarak terjauh yaitu 585 cm. Pengujian dilakukan sebanyak 10 kali.

3.4.2 Medan Pandang (Field Of View)

Pengujian ini dilakukan untuk mendapatkan medan pandang dari robot atau Field Of View. Medan pandang menjadi perlu diketahui sebagai acuan jangkauan kiri-kanan. Berdasarkan datasheet webcam Logitech C170 yang digunakan robot, diketahui bahwa Field Of View horizontal adalah 58 derajat.Untuk memvalidasinya dilakukan pengambilan gambar dengan sudut jangkauan horizontal 58 derajat.

3.4.3 Kecepatan

Pengujian kecepatan ini bertujuan untuk mengetahui pengaruh kecepatan terhadap gerak robot. Dalam hasil kecepatan ini ada tiga faktor dalam desain yang dapat mempengaruhi yaitu:

- Jenis motor. Jenis yang digunakan disini adalah motor DC dengan internal gir yang menghasilkan rpm maximal adalah 510 rpm dan 15kg.cm. Data didapatkan dari label yang tertera pada produk.
- Sumber daya. Motor yang digunakan disini memiliki kebutuhan tenaga optimal adalah 12 volt 1 A, sedangkan baterai yang digunakan untuk motor adalah LiPo 3 sel dengan tegangan 11,1 volt (3 x 3,7 volt) yang mampu mensuplai arus 1 A. Berdasarkan informasi ini maka motor tidak mengalami masalah dengan sumber daya.
- Selang waktu eksekusi motor. Dalam controller motor telah diatur selang waktu untuk bergeraknya motor hingga controller mengeksekusi perintah berikutnya. Untuk gerak

kanan-kiri disini diatur sebesar 50 ms sedangkan untuk maju adalah 250 ms.

- Lebar sinyal pulsa dari controller motor ke driver

Pengujian kecepatan dilakukan untuk mendapatkan rentang kecepatan yang dapat diatur pada robot pada lebar pulsa yang telah diatur. Motor robot secara default akan berputar dengan torsi 510 rpm dan torsi 15 kg.cm pada daya 12 volt 1A. Memberi daya kepada motor dalam pulsa dengan duty-cycle tertentu dapat mereduksi kecepatan motor agar sesuai dengan kebutuhan.

Pengujian dilakukan dalam 2 tahap yaitu

- Menguji kecepatan rotasi roda di setiap kenaikan 10% pulsa duty-cycle
- Menguji dengan run-test untuk mendekati bola pada jarak 90 cm

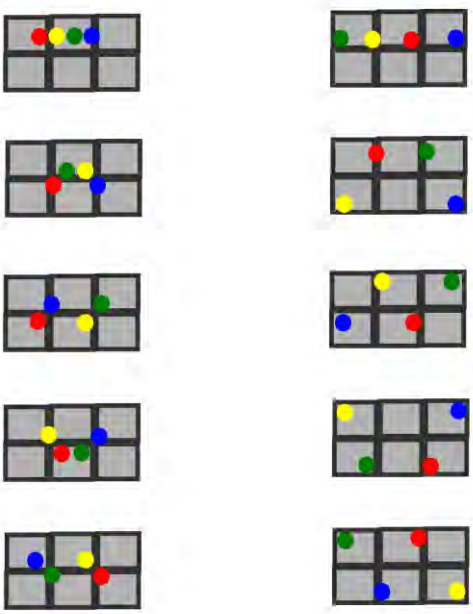
3.4.4 Pengaruh Pencahayaan

Pengujian ini ditujukan untuk menguji spesifikasi robot untuk melihat pengaruh tingkat pencahayaan pada pembacaan bola. Tingkat pencahayaan yang di ukur pada lumen 198, 74, 32, 17, 05 dan 0 (Gelap). Setiap tingkat cahaya di ambil 10 foto sehingga total didapat 60 foto yang akan disajikan selengkapnya dalam lampiran.

3.4.5 Pengujian Pemilihan warna

Pengujian ini ditujukan untuk menguji spesifikasi robot dalam memilih warna. Dalam pengujian ini, bola biru yang telah digunakan dalam pengujian-pengujian sebelumnya diletakkan berdekatan dengan tambahan 3 bola lain dengan warna merah, kuning dan hijau. Setiap bola terbuat dari bahan yang sama dengan ukuran yang sama pula (identik). Semua bola ditempatkan dalam posisi yang random dalam 6 petak persegi dengan konfigurasi acak. Posisi dibagi dalam 2 kategori yaitu 5 posisi bola saling berdekatan

dan 5 posisi memiliki jarak antar bola namun tetap dalam 6 persegi tadi.



Gambar 3.16 Variasi posisi bola

Gambar 3.16 menunjukkan 10 variasi tersebut.Untuk menguji robot dalam memilih warna biru, ke-empat bola tersebut diletakkan dalam 10 variasi posisi.

BAB IV
HASIL DAN PEMBAHASAN

Pada bab ini akan dijelaskan mengenai analisa data pada setiap pengujian spesifikasi. Pengujian yang telah dilakukan meliputi:

- Jarak terjauh
- Medan Pandang (Field Of View)
- Kecepatan
- Pencahayaan
- Pemilihan warna

4.1 Jarak terjauh

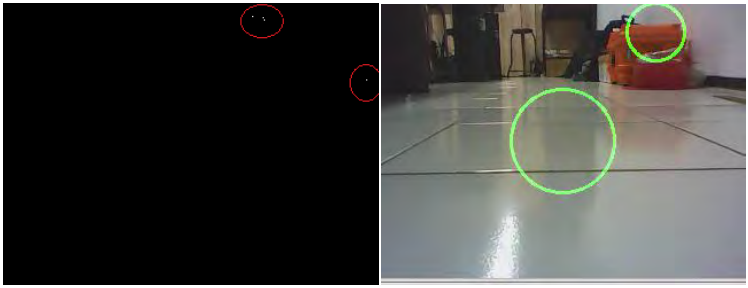
Pengujian ini adalah bertujuan untuk mengetahui jarak terjauh yang dapat dibaca oleh robot sehingga robot dapat menemukannya. Dalam pengujian dilakukan 10 kali run-test dan hasilnya robot mampu menemukan bola. Melalui data untuk penentuan nilai HSV diketahui bahwa pada rentang nilai HSV yang dipilih diketahui bahwa nilai pixel terkecil yang dapat terdeteksi adalah 31. Gambar 4.1 berikut adalah salah satu contoh gambar objek pada jarak terjauh (585 cm).

(a) (b)

Gambar 4.1 Contoh pembacaan pada jarak 585cm. Gambar (a) adalah pixel hasil threshold sedangkan gambar (b) adalah gambar hasil penentuan koordinat.

Maka pada gambar 4.1 pixel yang merepresentasikan bola adalah kumpulan pixel kecil yang berada sekitar koordinat pixel 20,240 dengan origin adalah pixel kiri atas. Namun dikarenakan adanya noise maka posisi lingkaran target tidak tepat pada koordinat tersebut. Metode penentuan dengan algoritma pusat massa pada 2 dimensi menyebabkan lingkaran target berada di antara kumpulan pixel bola dan kumpulan noise.

Sedangkan untuk hasil dimana tidak ada bola, didapat bahwa pixel hasil threshold adalah 12 pixel dengan tidak cenderung berkumpul pada area tertentu. Gambar 4.2 berikut menunjukkan hasil threshold tersebut.



(a) (b)

Gambar 4.2 Contoh pembacaan tanpa bola. Gambar (a) adalah pixel hasil threshold sedangkan gambar (b) adalah gambar hasil penentuan koordinat.

Gambar 4.2 adalah gambar asli tanpa adanya bola namun tetap lingkaran target memperoleh koordinat. Dikarenakan masih adanya pixel hasil threshold walaupun pixel tersebut adalah pixel yang bukan berasal dari bola, maka tetap akan muncul hasil penentuan koordiat.

Dari data jarak terjauh dan data bola kosong dapat diambil informasi bahwa:

- Jumlah pixel hasil threshold antara jarak terjauh dan tanpa bola tidak memiliki perbedaan jauh. Hal ini disebabkan karena ukuran proyeksi bola itu sendiri sudah mengecil dan terdapat noise.

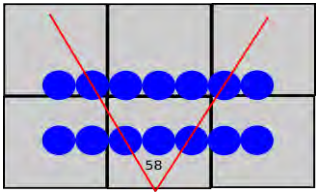
- Keadaan tanpa bola dan posisi bola terjauh tetap menghasilkan posisi koordinat karena adanya pixel noise, sedangkan pada posisi bola terjauh menghasil koordinat yang tidak tetap sehingga apabila robot mengambil koordinat tersebut sebagai acuan maka robot akan semakin menjauhi bola.

Berdasarkan informasi ini maka robot dapat salah dalam menemukan bola. Untuk mencegah gerak robot yang semakin menjauhi bola, maka ditentukan minimal pixel berdasarkan data yang menghasilkan koordinat target paling dekat dengan bola yaitu 31 pixel.

4.2 Medan Pandang

Penentuan Medan Pandang atau Field Of View (FOV) kamera kamera menjadi sangat penting karena dalam algoritma proses menemukan bola robot akan mengarahkan kamera hingga lingkaran target berada di masuk pada lingkaran tengah gambar. Jika dalam proses ini justru bola berada di luar FOV, maka robot tidak dapat menemukan lagi.

Nilai sudut FOV baik dari pengujian maupun data produk menunjukkan bahwa sudut FOV adalah 58 derajat. Gambar 4.3 adalah gambaran FOV pada sudut 58 derajat mempengaruhi tertangkapnya bola oleh kamera.



Gambar 4.3 Jangkauan FOV

Dari gambar 4.6 tersebut bahwa semakin bola memiliki jarak terhadap kamera maka bola akan lebih mudah tertangkap kamera walaupun ukurannya menjadi lebih kecil. Gambar 4.4 berikut menunjukkan hasil tangkapan kamera pada baris kedua yaitu jarak

30 cm. Dari hasil tersebut bola lebih banyak tertangkap kamera meskipun luasan bola lebih kecil.

Gambar 4.4 Hasil tangkapan pada baris kedua

Sedangkan untuk baris pertama, luasan bola lebih besar namun bola menjadi lebih sedikit tertangkap kamera. Gambar 4.5 menunjukkan hasil tangkapan bola pada baris pertama yaitu pada jarak 15 cm.

Gambar 4.5 Hasil tangkapan pada baris pertama

Berdasarkan hasil ini maka diketahui bahwa sebelum jarak 15 cm robot akan lebih mudah mengarahkan kamera agar robot menemukan bola dan mendekatinya. Namun setelah 15 cm atau kurang, robot akan lebih sulit untuk menemukan posisi bola dengan lingkaran target masuk ke lingkaran tengah gambar. Hal ini menjadi sebab pada saat running-test robot lbih sedikit bergerak memutar kiri-kanan untuk kemudian bergerak maju ketika jarak bola masih lebih dari 15cm. Namun ketika jarak telah kurang 15 cm, maka robot lebih banyak memutar kiri-kanan ketimbang ketika jarak kamera sebelum 15cm.

4.3 Kecepatan

Untuk dapat mereduksi kecepatan motor maka dilakukan metode pendayaan tidak kontinu kepada motor. Ukuran seberapa banyak tenaga yang disuplai setiap periodenya disebut duty-cycle. Periode yang digunakan disini adalah 1/100 detik. Telah dilakukan pengujian untuk mendapatkan nilai rpm pada setiap pengujian.

Apabila telah diketahui bahwa diameter roda adalah 45cm maka nilai setiap rpm ini dikalikan dengan waktu selang akan didapatkan jarak translasi untuk setiap eksekusi satu perintah. Dengan keliling roda adalah 283 cm maka jarak tempuh untuk gerak kiri-kanan dan gerak maju disajikan dalam tabel 4.1 berikut.

Tabel 4.1 Hubungan Duty-Cycle dan Jarak termpuh roda

Duty (%)	rpm	rps	Rotasi (cm)	Perdetik (cm)	Putar (cm)	Maju (cm)	Gagal
10	60	1.0	141.4	141.4	7.1	35.4	Ya
20	126	2.1	141.4	297.0	14.9	74.3	Tidak
30	147	2.5	141.4	346.5	17.3	86.6	Tidak
40	200	3.3	141.4	471.4	23.6	117.9	Tidak
50	341	5.7	141.4	803.8	40.2	200.9	Tidak
60	398	6.6	141.4	938.1	46.9	234.5	Tidak
70	398	6.6	141.4	938.1	46.9	234.5	Tidak
80	397	6.6	141.4	935.8	46.8	233.9	Tidak
90	445	7.4	141.4	1048.9	52.4	262.2	Tidak
100	510	8.5	141.4	1202.1	60.1	300.5	Ya

Berdasarkan tabel diketahui bahwa duty-cycle yang mulai 10% hingga 100% memiliki jangkauan perpindahan yang cukup jauh. Namun dalam pengujian nyata yaitu dengan running test diketahui hanya duty-cycle 20% hingga 90% yang dapat digunakan.

Supplai tenaga dengan duty-cycle secara perhitungan memang akan menyebabkan robot mampu berotasi dengan panjang perpindahan melingkar adalah 7cm dalam sekali perintah, namun supplai tenaga yang singkat ini telah membuat motor berkurang tidak hanya pada rpm namun juga torsi. Pada duty-cycle ini torsi sudah terlalu kecil untuk dapat menggerakkan robot.

Untuk duty-cycle 100% robot akan benar mendapatkan tenaga penuh sehingga berputar dengan 510 rpm dengan torsi maximal. Pada kondisi ini motor berputar berlebihan sehingga gagal mengarahkan lingkar target ke tengah gambar. Sedangkan pada duty-cycle 90% motor telah mengalami reduksi rpm dan torsi sehingga

mampu mengarahkan lingkar target ke tengah gambar tanpa mengalami kegagalan.

4.4 Pencahayaan

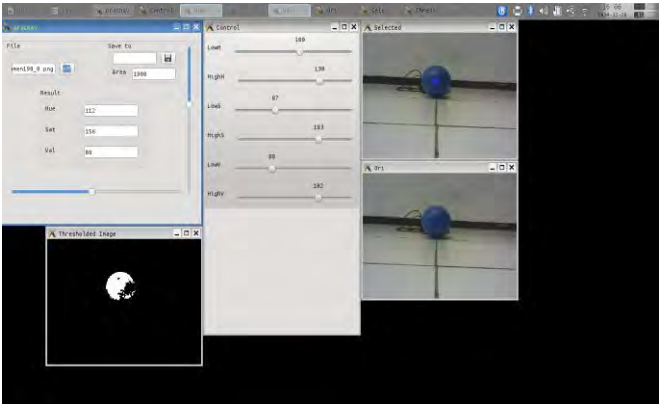
Pengujian ini bertujuan untuk mengetahui rentang tingkat pencahayaan terhadap pembacaan robot. Pengukuran dilakukan dengan variasi cahaya 198 lux, 74 lux, 32 lux, 17 lux, 5 lux dan 0 lux (Gelap). Masing-masing variasi di ambil 10 foto sehingga total di dapat 60 foto. Data tersebut selengkapnya akan disajikan dalam lampiran. Tabel 4.2 berikut menyajikan rata-rata hasil pembacaan pixel.

Tabel 4.2 Pixel rata-rata di tingkat pencahayaan berbeda.

Lux	Pixel	Gagal
198	1900	Tidak
74	1487	Tidak
32	1530	Tidak
17	1546	Tidak
5	936	Tidak
0	0	Ya

Untuk analisa lebih lanjut digunakanlah program untuk membaca sebuah pixel pada bola yang dipilih acak untuk diketahui nilai HSV yang dimiliki pixel tersebut. Gambar 4.6 berikut adalah contoh analisa menggunakan software tersebut untuk mendapatkan nilai HSV yang ditandai lingkaran biru. Pixel yang dipilih untuk di lihat nilai HSV nya adalah pixel yang diperkirakan merupakan centroid dengan melihat posisi pusat bola pada gambar secara manual.

Jumlah total pixel yang terbentuk menggunakan thresholding dengan rentang nilai HSV yang telah ditentukan sebelumnya sehingga apabila terjadi perubahan nilai HSV pada hasil tangkap kamera maka akan ada pixel yang tidak cocok dalam rentang thresholding. Dari tabel 4.2 terlihat semakin sedikitnya pixel yang tidak cocok dengan rentang thresholding HSV.



Gambar 4.6 Salah satu analisa HSV

Tabel 4.3 berikut ini berisi nilai pixel HSV untuk pixel yang sama (ditandai lingkaran biru) pada setiap tingkat pencahayaan.

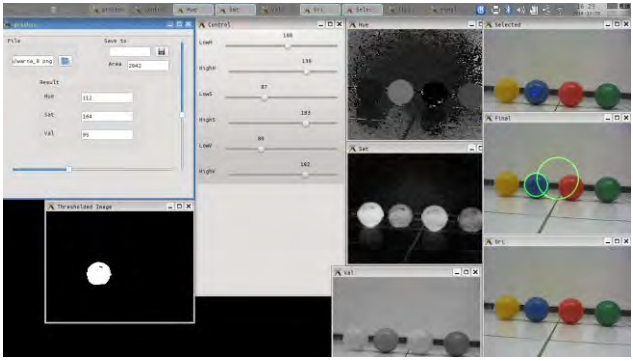
Lumen	Hue	Saturation	Value
198	112	156	88
74	112	139	79
32	112	161	90
17	112	157	94
5	112	115	73
0	0	0	30

Berdasarkan data di atas diketahui bahwa nilai Hue cenderung tetap yaitu pada nilai 112 kecuali pada lumen 0. Sedangkan yang mengalami perubahan adalah nilai saturasi dan value. Seperti diketahui bahwa saturasi adalah tingkat suatu warna terhadap warna putih sehingga semakin gelapnya warna akan menurunkan tingkat saturasi, sedangkan value tingkat suatu warna terhadap warna hitam sehingga semakin cerahnya warna akan menurunkan tingkat value

4.5 Pemilihan Warna.

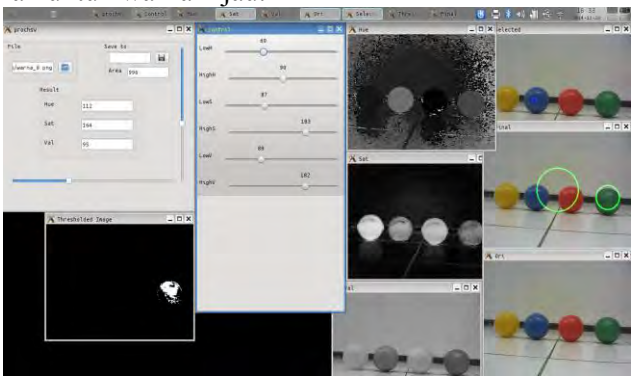
Pengujian ini bertujuan untuk menguji robot dalam hal

memilih warna. Disini digunakan 4 bola dengan warna berbeda yaitu merah, kuning, hijau, dan biru. Keempat bola tersebut memiliki ukuran yang sama. Untuk menganalisa digunakan program seperti pada gambar 4.10. Dengan mengambil nilai HSV yang telah ditentukan di dapatkan lingkaran target memilih warna biru.



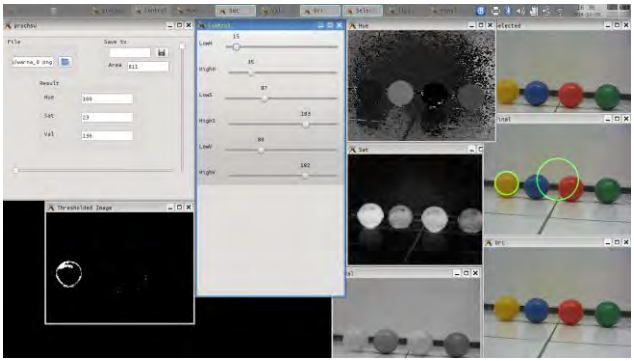
Gambar 4.7 Analisa seleksi warna biru

Kemudian dengan software tersebut dengan tetap menggunakan jangkauan Saturation dan Value dilakukan pengaturan untuk mendapatkan warna hijau. Gambar 4.11 adalah hasil pengaturan untuk warna hijau.



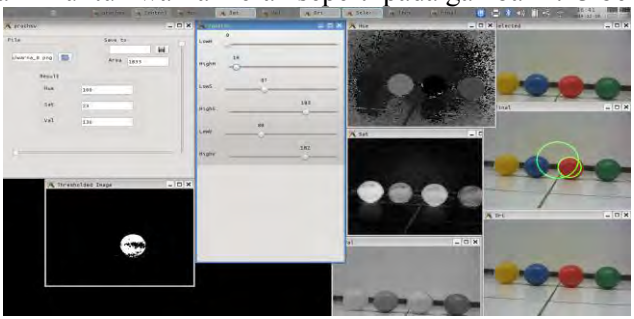
Gambar 4.8 Analisa seleksi warna hijau

Selanjutnya untuk warna kuning seperti pada gambar 4.12



Gambar 4.9 Analisa seleksi warna hijau

Dan terakhir untuk warna merah seperti pada gambar 4.13 berikut.



Gambar 4.10 Analisa seleksi warna hijau

Berdasarkan analisa di atas diketahui rentang Hue setiap bola adalah Biru pada 100-130, Hijau pada 60-90, Kuning pada 15-35, dan Merah pada 0-10.Dengan rentang nilai Hue yang berbeda-beda di atas maka dengan mudah dapat dibedakan setiap bola.

Halaman ini memang sengaja dikosongkan

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan perancangan dan pembangunan robot vision dengan perangkat RaspberryPi ini dapat diambil kesimpulan yaitu:

- Robot hasil perancangan dan pembangunan dapat menemukan objek berdasarkan warna yang telah ditetapkan diantara warna-warna lain.
- Rentang nilai yang dapat digunakan adalah untuk saturasi adalah 87-183, sedangkan untuk value adalah 80-182. Untuk nilai Hue bergantung oleh warna objek,sedangkan untuk objek uji disini yang berwarna biru adalah 100-130.
- Posisi jarak terjauh bola uji berdiameter 97.8 mm yang dapat dibaca oleh robot adalah 585 cm
- Kecepatan putaran roda yang dapat digunakan adalah minimal 126 rpm pada duty-cycle 20% dan maximal adalah 445 rpm pada duty-cycle 90%
- Tingkat pencahayaan minimal adalah 5 lumen karena tingkat pencahayaan sangat berpengaruh terhadap nilai Saturation dan Value.

5.2 Saran

Untuk pengembangan lebih lanjut maka disarankan dilakukan beberapa hal:

- Sistem komputer tertanam tidak menggunakan RaspberryPi namun sistem lebih tinggi seperti PCDuino
- Kamera menggunakan merek kamera lain dengan resolusi lebih tinggi dan proses lebih cepat. Untuk tujuan khusus dapat digunakan kamera khusus seperti kamera IR atau multispektral.
- Desain Robot dapat menambah akuator seperti tangan atau kaki
- Algoritma dapat ditambahkan dengan Clustering, Countour, atau Pattern Recognizing berbasis jaringan saraf tiruan.

Halaman ini memang sengaja dikosongkan

LAMPIRAN A: DATA PENENTUAN UNTUK PENENTUAN NILAI SATURATION DAN VALUE

Nomor	Jarak	S min	S max	V min	V max	Area Threshold
1	0	92	255	80	208	64535
2		92	255	80	208	63016
3		92	255	80	208	63815
4		92	255	80	208	64445
5		92	255	80	208	63617
6		92	255	80	208	63948
7		92	255	80	208	63829
8		92	255	80	208	63132
9		92	255	80	208	63163
10		92	255	80	208	62462
11	15	53	190	54	169	33730
12		77	190	70	169	33698
13		10	176	13	176	34209
14		4	242	5	202	34752
15		30	214	37	219	34259
16		5	219	12	232	33458
17		0	223	35	231	33294
18		0	223	13	231	33974
19		0	223	0	231	37160
20		24	229	23	231	36903
21	30	88	181	62	182	7103
22		88	181	62	182	7039
23		88	181	62	183	6994
24		73	181	63	183	7202
25		73	181	69	183	6039
26		73	181	69	183	6037
27		73	181	36	184	7851
28		73	167	36	153	7861
29		81	167	36	153	8027
30		81	167	36	192	8066
31	45	78	193	36	206	4337

Nomor	Jarak	S min	S max	V min	V max	Area Threshold
32		78	193	36	180	4329
33		78	182	44	201	4315
34		82	182	34	202	4264
35		73	199	53	178	4386
36		73	199	54	178	4358
37		73	204	54	183	4377
38		73	204	54	189	4377
39		73	204	27	159	4446
40		73	204	27	159	4438
41	60	75	190	53	168	2247
42		75	190	40	168	2300
43		53	190	40	191	2467
44		53	173	47	191	2441
45		53	173	46	191	2461
46		66	180	44	180	2383
47		66	180	44	180	2390
48		66	180	40	160	2377
49		66	180	44	163	2357
50		71	180	48	172	2310
51	75	70	180	41	180	1650
52		70	180	41	167	1659
53		70	180	48	167	1656
54		103	180	40	167	1350
55		103	180	40	167	1381
56		103	180	40	183	1366
57		103	180	38	183	1374
58		103	180	47	196	1338
59		103	180	47	196	1378
60		103	180	48	187	1348
61	90	84	180	57	179	1051
62		84	180	57	179	1047
63		84	180	51	179	1088
64		84	180	51	179	1064

Nomor	Jarak	S min	S max	V min	V max	Area Threshold
65		84	180	51	179	1078
66		84	180	51	179	1097
67		84	180	60	179	1044
68		84	180	60	179	1040
69		84	180	60	179	1039
70		84	180	60	179	1035
71	105	58	187	66	201	843
72		58	187	66	201	826
73		58	187	66	201	823
74		58	187	66	201	842
75		58	187	66	201	822
76		58	187	66	201	845
77		58	187	66	201	838
78		58	187	66	201	834
79		58	187	66	201	844
80		58	187	66	201	821
81	120	77	180	76	180	567
82		77	180	76	180	568
83		77	180	76	180	566
84		77	180	78	180	564
85		77	180	78	180	560
86		77	180	78	180	557
87		77	180	78	180	562
88		77	180	78	180	561
89		77	180	78	180	551
90		77	180	78	180	560
91	135	70	180	80	180	484
92		70	180	80	180	473
93		70	180	80	180	473
94		70	180	80	180	478
95		70	180	80	180	483
96		70	180	80	180	486
97		70	180	80	180	488

Nomor	Jarak	S min	S max	V min	V max	Area Threshold
98		70	180	80	180	482
99		70	180	80	180	471
100		70	180	80	180	475
101	150	70	180	75	180	387
102		70	180	75	180	386
103		70	180	75	180	397
104		70	180	75	180	386
105		70	180	75	180	382
106		70	180	75	180	376
107		70	180	75	180	390
108		70	180	75	180	394
109		70	180	75	180	392
110		70	180	75	180	387
111	165	102	180	66	180	305
112		102	180	66	180	295
113		102	180	66	180	302
114		102	180	66	180	305
115		102	180	66	180	305
116		102	180	66	180	321
117		102	180	66	180	304
118		102	180	66	180	304
119		102	180	66	180	310
120		102	180	66	180	299
121	180	78	180	73	180	279
122		78	180	73	180	272
123		78	180	73	180	266
124		78	180	73	180	265
125		78	180	73	180	276
126		78	180	73	180	272
127		78	180	73	180	263
128		78	180	73	180	271
129		78	180	73	180	256
130		78	180	73	180	271

Nomor	Jarak	S min	S max	V min	V max	Area Threshold
131	195	88	180	93	180	201
132		88	180	93	180	203
133		88	180	93	180	199
134		88	180	93	180	196
135		88	180	93	180	196
136		88	180	93	180	197
137		88	180	93	180	193
138		88	180	93	180	191
139		88	180	93	180	193
140		88	180	93	180	188
141	210	80	180	93	191	158
142		80	180	93	191	167
143		80	180	93	191	157
144		80	180	93	191	162
145		80	180	93	191	162
146		80	180	93	191	162
147		80	180	93	191	157
148		80	180	93	191	169
149		80	180	93	191	163
150		80	180	93	191	168
151	225	70	180	85	180	172
152		70	180	85	180	176
153		70	180	85	180	171
154		70	180	85	180	169
155		70	180	85	180	171
156		70	180	85	180	173
157		70	180	85	180	176
158		70	180	85	180	179
159		70	180	85	180	168
160		70	180	85	180	170
161	240	81	180	101	180	109
162		81	180	101	180	91
163		81	180	101	180	109

Nomor	Jarak	S min	S max	V min	V max	Area Threshold
164		81	180	101	180	95
165		81	180	101	180	97
166		81	180	101	180	108
167		81	180	101	180	98
168		81	180	101	180	102
169		81	180	101	180	93
170		81	180	101	180	116
171	255	86	180	70	180	149
172		86	180	70	180	153
173		86	180	70	180	144
174		86	180	70	180	145
175		86	180	70	180	143
176		86	180	70	180	140
177		86	180	70	180	142
178		86	180	70	180	156
179		86	180	70	180	141
180		86	180	70	180	141
181	270	70	180	93	180	107
182		70	180	93	180	104
183		70	180	93	180	103
184		70	180	93	180	104
185		70	180	93	180	102
186		70	180	93	180	105
187		70	180	93	180	102
188		70	180	93	180	103
189		70	180	93	180	110
190		70	180	93	180	101
191	285	70	180	71	180	120
192		70	180	71	180	120
193		70	180	71	180	127
194		70	180	71	180	119
195		70	180	71	180	123
196		70	180	71	180	119

Nomor	Jarak	S min	S max	V min	V max	Area Threshold
197		70	180	71	180	120
198		70	180	71	180	122
199		70	180	71	180	118
200		70	180	71	180	124
201	300	107	180	81	180	80
202		107	180	81	180	88
203		107	180	81	180	85
204		107	180	81	180	82
205		107	180	81	180	81
206		107	180	81	180	81
207		107	180	81	180	81
208		107	180	81	180	85
209		107	180	81	180	84
210		107	180	81	180	91
211	315	70	180	71	180	87
212		70	180	71	180	92
213		70	180	71	180	96
214		70	180	71	180	95
215		70	180	71	180	91
216		70	180	71	180	95
217		70	180	71	180	89
218		70	180	71	180	95
219		70	180	71	180	90
220		70	180	90	180	77
221	330	70	180	88	180	70
222		70	180	88	180	73
223		70	180	88	180	66
224		70	180	88	180	71
225		70	180	88	180	73
226		70	180	88	180	70
227		70	180	88	180	78
228		70	180	88	180	77
229		70	180	88	180	73

Nomor	Jarak	S min	S max	V min	V max	Area Threshold
230		70	180	88	180	67
231	345	70	180	88	180	66
232		70	180	88	180	72
233		70	180	88	180	69
234		70	180	88	180	65
235		70	180	88	180	71
236		70	180	88	180	72
237		70	180	88	180	71
238		70	180	88	180	70
239		70	180	88	180	68
240		70	180	88	180	71
241	360	70	180	114	180	8
242		70	180	114	180	11
243		70	180	114	180	14
244		70	180	114	180	13
245		70	180	114	180	14
246		70	180	114	180	15
247		70	180	114	180	15
248		70	180	114	180	12
249		70	180	114	180	14
250		70	180	114	180	12
251	375	85	197	99	177	38
252		85	197	99	177	36
253		85	197	99	177	43
254		85	197	99	177	36
255		85	197	99	177	43
256		85	197	99	177	36
257		85	197	99	177	35
258		85	197	99	177	36
259		85	197	99	177	44
260		85	197	99	177	36
261	390	70	180	94	180	41
262		70	180	95	180	40

Nomor	Jarak	S min	S max	V min	V max	Area Threshold
263		70	180	95	180	37
264		70	180	95	180	44
265		70	180	95	180	39
266		70	180	95	180	43
267		70	180	95	180	39
268		70	180	95	180	41
269		70	180	95	180	42
270		70	180	95	180	41
271	405	101	180	101	180	37
272		101	180	101	180	36
273		101	180	101	180	32
274		101	180	101	180	36
275		101	180	101	180	37
276		101	180	101	180	37
277		101	180	101	180	38
278		101	180	101	180	33
279		101	180	101	180	34
280		101	180	101	180	35
281	420	87	180	100	180	35
282		87	180	100	180	34
283		87	180	100	180	35
284		87	180	100	180	35
285		87	180	100	180	36
286		87	180	100	180	34
287		87	180	100	180	38
288		87	180	100	180	39
289		87	180	100	180	40
290		87	180	100	180	33
291	435	92	180	89	180	39
292		92	180	89	180	37
293		92	180	89	180	43
294		92	180	89	180	34
295		114	180	89	180	34

Nomor	Jarak	S min	S max	V min	V max	Area Threshold
296		114	180	89	180	31
297		114	180	89	180	32
298		114	180	89	180	32
299		114	180	89	180	31
300		114	180	89	180	31
301	450	130	180	61	190	26
302		130	180	61	190	25
303		130	180	61	190	22
304		130	180	61	190	28
305		130	180	61	190	27
306		130	180	61	190	32
307		130	180	61	190	22
308		130	180	61	190	27
309		130	180	61	190	25
310		130	180	61	190	26
311		101	180	100	180	25
312	465	101	180	100	180	24
313		101	180	100	180	24
314		101	180	100	180	27
315		101	180	100	180	24
316		101	180	100	180	22
317		101	180	100	180	23
318		101	180	100	180	23
319		101	180	100	180	26
320		101	180	100	180	26
321	480	113	180	95	180	20
322		113	180	95	180	23
323		113	180	95	180	21
324		113	180	95	180	26
325		113	180	95	180	25
326		113	180	95	180	24
327		113	180	95	180	25
328		113	180	95	180	25

Nomor	Jarak	S min	S max	V min	V max	Area Threshold
329		113	180	95	180	23
330		113	180	95	180	26
331	495	141	180	100	180	9
332		141	180	100	180	7
333		141	180	100	180	7
334		141	180	100	180	7
335		141	180	100	180	15
336		141	180	100	180	13
337		141	180	100	180	7
338		141	180	100	180	10
339		141	180	100	180	13
340		141	180	100	180	12
341	510	70	180	96	180	22
342		70	180	96	180	21
343		70	180	96	180	22
344		70	180	96	180	26
345		70	180	96	180	27
346		70	180	96	180	30
347		70	180	96	180	22
348		70	180	96	180	20
349		70	180	96	180	21
350		70	180	96	180	29
351	525	112	180	80	180	22
352		112	180	80	180	24
353		112	180	80	180	20
354		112	180	80	180	17
355		112	180	80	180	21
356		112	180	80	180	22
357		112	180	80	180	21
358		112	180	80	180	21
359		112	180	80	180	19
360		112	180	80	180	21
361	540	111	180	70	180	24

Nomor	Jarak	S min	S max	V min	V max	Area Threshold
362		111	180	70	180	21
363		111	180	70	180	29
364		111	180	70	180	22
365		111	180	70	180	26
366		111	180	70	180	22
367		111	180	70	180	21
368		111	180	70	180	24
369		111	180	70	180	24
370		111	180	70	180	23
371	555	82	193	96	202	18
372		82	193	85	202	27
373		92	193	90	202	20
374		92	193	90	202	22
375		92	193	90	202	19
376		92	193	90	202	18
377		161	193	72	202	1
378		84	193	95	202	12
379		84	193	95	202	16
380		84	193	95	202	13
381	570	87	180	70	180	25
382		87	180	70	180	31
383		81	180	92	180	15
384		81	180	92	180	18
385		81	180	92	180	12
386		81	180	92	180	15
387		81	180	92	180	17
388		81	180	92	180	17
389		64	180	89	180	26
390		106	180	64	180	28
391	595	70	180	101	180	15
392		93	180	101	180	6
393		93	180	101	180	12
394		93	180	101	180	13

Nomor	Jarak	S min	S max	V min	V max	Area Threshold
395		93	180	101	180	13
396		93	180	101	180	8
397		93	180	101	180	10
398		93	180	101	180	11
399		132	180	101	180	3
400		46	180	76	180	34
401	610	70	180	70	180	28
402		106	180	97	180	7
403		106	180	97	180	6
404		106	180	97	180	14
405		106	180	97	180	12
406		106	180	97	180	15
407		106	180	97	180	7
408		106	180	97	180	9
409		106	180	97	180	13
410		101	180	70	180	23
411	625	124	180	96	180	6
412		124	180	96	180	7
413		124	180	96	180	9
414		124	180	96	180	8
415		124	180	96	180	8
416		124	180	96	180	6
417		124	180	96	180	9
418		124	180	96	180	8
419		124	180	96	180	4
420		124	180	96	180	8
421	640	103	180	83	180	19
422		103	180	83	180	19
423		103	180	83	180	24
424		103	180	83	180	18
425		103	180	83	180	17
426		103	180	83	180	20
427		103	180	83	180	21

Nomor	Jarak	S min	S max	V min	V max	Area Threshold
428		103	180	83	180	19
429		126	180	83	180	15
430		126	180	83	180	13
431	665	123	180	70	180	12
432		123	180	87	180	10
433		123	180	87	180	11
434		123	180	87	180	12
435		123	180	87	180	12
436		139	180	80	180	3
437		139	180	80	180	13
438		139	180	80	180	7
439		139	180	80	180	6
440		139	180	80	180	7

Rata-Rata:
Smin = 87
Smax = 183
Vmin = 80
Vmax = 182

LAMPIRAN B: KODE SUMBER C++ UNTUK PERANGKAT LUNAK ROBOT.

Seluruh file ini dapat di akses di URL:
https://github.com/mekatronik-achmadi/tugas_akhir/tree/master/software/qrobotvision3

Seluruh kode sumber ini dibangun dengan operating system ubuntu atau debian. Kode sumber terdiri dari file:

1. qrobot.pro
2. qrobot.cpp
3. qrobot.h
4. qrobot.ui
5. main.cpp
6. SerialPort/

Untuk dapat mengkompilasi kode sumber ini maka dibutuhkan paket berikut:

1. libopencv-dev
2. libqt4-dev
3. build-essential

Karena berbasis qt4 maka untuk mengkompilasi dapat menggunakan program qmake dan make. Untuk memanggilnya dibutuhkan privilage root agar dapat mengakses /dev/tty*

Untuk file di dalam folder SerialPort adalah file pustaka QextSerialPort yang digunakan tanpa ada perubahan apapun.

1. File qrobot.pro

```
#-------------------------------------------------
#
# Project created by QtCreator 2014-10-21T09:39:07
#
#-------------------------------------------------

QT       += core gui

greaterThan(QT_MAJOR_VERSION, 4): QT += widgets

TARGET = qrobot
TEMPLATE = app
```

```
SOURCES += main.cpp\
        qrobot.cpp \
    SerialPortLibs/posix_qextserialport.cpp \
    SerialPortLibs/qextserialbase.cpp \
    SerialPortLibs/qextserialenumerator.cpp \
    SerialPortLibs/qextserialport.cpp

HEADERS  += qrobot.h \
    SerialPortLibs/posix_qextserialport.h \
    SerialPortLibs/qextserialbase.h \
    SerialPortLibs/qextserialenumerator.h \
    SerialPortLibs/qextserialport.h

FORMS    += qrobot.ui

DEPENDPATH += .
INCLUDEPATH += .
INCLUDEPATH += /usr/local/include/opencv2
LIBS += -lopencv_core -lopencv_highgui -
lopencv_imgproc

DEFINES += _TTY_POSIX_
```

2. File qrobot.cpp

```
#include "qrobot.h"
#include "ui_qrobot.h"

int iLowH = 100;
int iHighH = 130;

int iLowS = 87;
int iHighS = 183;

int iLowV = 80;
int iHighV = 182;

uint minArea = 31;
uint maxArea = 22500;

qrobot::qrobot(QWidget *parent) :
    QMainWindow(parent),my_port(0),
    ui(new Ui::qrobot)
{
```

```
    ui->setupUi(this);

    my_cam_timer = new QTimer(this);


QObject::connect(my_cam_timer,SIGNAL(timeout()),this,SLO
T(img_proc()));
    my_cam_timer->start(10);

    cam.open(0);

    if ( !cam.isOpened()){
        std::cout << "Cannot open the web cam" <<
std::endl;
    }

    cv::namedWindow("Control", CV_WINDOW_AUTOSIZE);

    cv::createTrackbar("LowH", "Control", &iLowH, 179);
    cv::createTrackbar("HighH", "Control", &iHighH,
179);

    cv::createTrackbar("LowS", "Control", &iLowS, 255);
    cv::createTrackbar("HighS", "Control", &iHighS,
255);

    cv::createTrackbar("LowV", "Control", &iLowV, 255);
    cv::createTrackbar("HighV", "Control", &iHighV,
255);

    cam.set(CV_CAP_PROP_FRAME_WIDTH, 320);
    cam.set(CV_CAP_PROP_FRAME_HEIGHT, 240);

    this->setWindowTitle("QRobot [Closed]");
    ui->actionOpen->setEnabled(true);
    ui->actionClose->setEnabled(false);
    ui->actionStop->setEnabled(false);
    ui->actionFoward->setEnabled(false);
    ui->actionBackward->setEnabled(false);
    ui->actionRight->setEnabled(false);
    ui->actionLeft->setEnabled(false);


    qrobot::on_actionOpen_triggered();

    my_timer = new QTimer(this);
```

```
QObject::connect(my_timer,SIGNAL(timeout()),this,SLOT(bo
oted()));
    my_timer->start(500);

    my_move = new QTimer(this);


QObject::connect(my_move,SIGNAL(timeout()),this,SLOT(mov
e_proc()));
    my_move->start(500);


QObject::connect(my_port,SIGNAL(readyRead()),this,SLOT(n
ext_move()));
}

qrobot::~qrobot()
{
    delete ui;
}

void qrobot::on_actionOpen_triggered()
{
    if(my_port){

        if(my_port->isOpen()){
            my_port->close();
        }

        delete my_port;
        my_port=NULL;
    }

    QString my_device= "/dev/ttyUSB0";
    BaudRateType my_baud = BAUD38400;

    my_port = new
QextSerialPort(my_device,QextSerialPort::Polling);
    my_port->setBaudRate(my_baud);
    my_port->setDataBits(DATA_8);
    my_port->setParity(PAR_NONE);
    my_port->setStopBits(STOP_1);
    my_port->setFlowControl(FLOW_OFF);
    my_port->setTimeout(100);
```

```
    if(!my_port->open(QIODevice::ReadWrite)){
        delete my_port;
        my_port=NULL;

        QString s="Cannot open device at ";
        s += my_device;
        QMessageBox::critical(this,"Error",s);

        this->setWindowTitle("QRobot [Closed]");
        ui->actionOpen->setEnabled(true);
        ui->actionClose->setEnabled(false);
        ui->actionStop->setEnabled(false);
        ui->actionFoward->setEnabled(false);
        ui->actionBackward->setEnabled(false);
        ui->actionRight->setEnabled(false);
        ui->actionLeft->setEnabled(false);
    }
    else{
        this->setWindowTitle("QRobot [Opened]");
        ui->actionOpen->setEnabled(false);
        ui->actionClose->setEnabled(true);
        ui->actionStop->setEnabled(true);
        ui->actionFoward->setEnabled(true);
        ui->actionBackward->setEnabled(true);
        ui->actionRight->setEnabled(true);
        ui->actionLeft->setEnabled(true);
    }
}

void qrobot::on_actionClose_triggered()
{
    if(my_port){

        if(my_port->isOpen()){
            my_port->close();
        }

        delete my_port;
        my_port=NULL;

        this->setWindowTitle("QRobot [Closed]");
        ui->actionOpen->setEnabled(true);
        ui->actionClose->setEnabled(false);
        ui->actionStop->setEnabled(false);
        ui->actionFoward->setEnabled(false);
```

```
        ui->actionBackward->setEnabled(false);
        ui->actionRight->setEnabled(false);
        ui->actionLeft->setEnabled(false);
    }
}

void qrobot::on_actionFoward_triggered()
{
    my_timer->stop();
    QByteArray comdata="foward\n";
    my_port->write(comdata);
    my_timer->start(500);
}

void qrobot::on_actionBackward_triggered()
{
    my_timer->stop();
    QByteArray comdata="backward\n";
    my_port->write(comdata);
    my_timer->start(500);
}

void qrobot::on_actionRight_triggered()
{
    my_timer->stop();
    QByteArray comdata="right\n";
    my_port->write(comdata);
    my_timer->start(500);
}

void qrobot::on_actionLeft_triggered()
{
    my_timer->stop();
    QByteArray comdata="left\n";
    my_port->write(comdata);
    my_timer->start(500);
}

void qrobot::on_actionStop_triggered()
{
    my_timer->stop();
    QByteArray comdata="stop\n";
    my_port->write(comdata);
    my_timer->start(500);
}
```

```
void qrobot::booted()
{
    my_timer->stop();
    QByteArray comdata="booted\n";
    my_port->write(comdata);
    my_timer->start(500);
}

void qrobot::img_proc()
{
    cv::Mat imgOriginal;

    bool bSuccess = cam.read(imgOriginal);
    if (!bSuccess) {
      std::cout << "Cannot read a frame from video
stream" << std::endl;
    }

    cv::Mat imgsqr= cv::Mat::zeros( imgOriginal.size(),
CV_8UC3 );

    cv::Mat imgHSV;
    cv::cvtColor(imgOriginal, imgHSV,
cv::COLOR_BGR2HSV);

    cv::Mat imgThresholded;
    cv::inRange(imgHSV, cv::Scalar(iLowH, iLowS, iLowV),
cv::Scalar(iHighH, iHighS, iHighV), imgThresholded);
    cv::imshow("Thresholded Image", imgThresholded);

    imgRows = imgThresholded.rows;
    imgCols = imgThresholded.cols;

    imgBiner =
cv::Mat(imgRows,imgCols,CV_8U,cv::Scalar(0));

    for(i=0;i<imgRows;i++){
        for(j=0;j<imgCols;j++){

if(imgThresholded.at<uchar>(i,j)==255){imgBiner.at<uchar
>(i,j)=1;}
            else{imgBiner.at<uchar>(i,j)=0;}
        }
    }

    xval=cv::Mat(1,imgCols,CV_16U,cv::Scalar(0));
```

```
    for(i=0;i<imgCols;i++){
        xval.at<ushort>(0,i)=0;
        for(j=0;j<imgRows;j++){

xval.at<ushort>(0,i)=xval.at<ushort>(0,i)+imgBiner.at<uc
har>(j,i);
        }
    }

    mArea=cv::Mat(1,1,CV_16U,cv::Scalar(0));
    for(i=0;i<imgCols;i++){

mArea.at<ushort>(0,0)=mArea.at<ushort>(0,0)+xval.at<usho
rt>(0,i);
    }
    Area=mArea.at<ushort>(0,0);

    if((Area>minArea)&&(Area<=maxArea)){

        mxSum=cv::Mat(1,1,CV_32F,cv::Scalar(0));
        for(i=0;i<imgCols;i++){

mxSum.at<float>(0,0)=mxSum.at<float>(0,0)+(i*xval.at<ush
ort>(0,i));
        }
        xSum=mxSum.at<float>(0,0);

        xcen=xSum/Area;

        yval=cv::Mat(1,imgRows,CV_16U,cv::Scalar(0));
        for(i=0;i<imgRows;i++){
            yval.at<ushort>(0,i)=0;
            for(j=0;j<imgCols;j++){

yval.at<ushort>(0,i)=yval.at<ushort>(0,i)+imgBiner.at<uc
har>(i,j);
            }
        }

        mySum=cv::Mat(1,1,CV_32F,cv::Scalar(0));

        for(i=0;i<imgRows;i++){

mySum.at<float>(0,0)=mySum.at<float>(0,0)+(i*yval.at<ush
ort>(0,i));
        }
```

```
        ySum=mySum.at<float>(0,0);

        ycen=ySum/Area;

    }

    imgCen = cv::Mat( imgOriginal.size(),
CV_8UC3,cv::Scalar(0,0,0));
    vCen=25;

cv::circle(imgCen,cv::Point(xcen,ycen),vCen,cv::Scalar(0
,255,0),2);

    imgRad= cv::Mat( imgOriginal.size(),
CV_8UC3,cv::Scalar(0,0,0));
    vRad=45;

cv::circle(imgRad,cv::Point(imgCols/2,imgRows/2),vRad,cv
::Scalar(0,255,0),2);

    cv::imshow("Thresholded Image", imgThresholded);
    imgFinal = imgOriginal + imgRad+imgCen;
    cv::imshow("Final", imgFinal);
}

void qrobot::move_proc(){
    xPos= xcen;
    yPos= xcen;

    xDef= imgCols/2;
    yDef= imgRows/2;

    vDef = 25;
    vArea= Area;

    if((Area>minArea)&&(Area<=maxArea)){

      std::cout << "xPos= " << xPos << " " << "yPos= "
<< yPos << " " << "Area= " << vArea << std::endl;

      if(xPos<xDef){
      if((xDef-xPos)>vDef){
          qrobot::on_actionLeft_triggered();
          std::cout << "LEFT" << std::endl;
      }
```

```
      else{
          qrobot::on_actionFoward_triggered();
          std::cout << "FOWARD" << std::endl;
      }
      }
      else if(xPos>xDef){
      if((xPos-xDef)>vDef){
          qrobot::on_actionRight_triggered();
          std::cout << "RIGHT" << std::endl;
      }
      else{
          qrobot::on_actionFoward_triggered();
          std::cout << "FOWARD" << std::endl;
      }
      }
      else if(xPos==xDef){
      qrobot::on_actionFoward_triggered();
      std::cout << "FOWARD" << std::endl;
      }
    }
    else{
      std::cout << "xPos= " << xPos << " " << "yPos= "
<< yPos << " " << "Area= " << vArea << std::endl;
      std::cout << "STOP" << std::endl;
    }
}
```

3. File qrobot.h

```
#ifndef QROBOT_H
#define QROBOT_H

#include <QMainWindow>
#include <QMessageBox>
#include <QTimer>
#include "SerialPortLibs/qextserialport.h"
#include "SerialPortLibs/qextserialenumerator.h"

#include <iostream>
#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/imgproc/imgproc.hpp>

namespace Ui {
class qrobot;
}
```

```
class qrobot : public QMainWindow
{
    Q_OBJECT

public:
    explicit qrobot(QWidget *parent = 0);
    ~qrobot();

private slots:
    void on_actionOpen_triggered();

    void on_actionClose_triggered();

    void on_actionFoward_triggered();

    void on_actionBackward_triggered();

    void on_actionRight_triggered();

    void on_actionLeft_triggered();

    void on_actionStop_triggered();

    void booted();

    void img_proc();

    void move_proc();

private:
    Ui::qrobot *ui;
    QextSerialPort *my_port;
    QTimer* my_cam_timer;
    QTimer* my_timer;

    cv::VideoCapture cam;

    uint imgRows;
    uint imgCols;

    uint i,j;

    cv::Mat imgBiner;

    uint sqr;
```

```
        cv::Mat mArea;
        uint Area;

        cv::Mat xval;
        cv::Mat mxSum;
        float xSum;
        uint xcen;

        cv::Mat yval;
        cv::Mat mySum;
        float ySum;
        uint ycen;

        cv::Mat imgRad;
        uint vRad;

        cv::Mat imgCen;
        uint vCen;

        cv::Mat imgFinal;

        QTimer* my_move;
        uint xPos,yPos;
        uint xDef,yDef;
        uint vDif,vDef;
        uint vArea;
};

#endif // QROBOT_H
```

4. File qrobot.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>qrobot</class>
 <widget class="QMainWindow" name="qrobot">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>442</width>
    <height>88</height>
   </rect>
  </property>
  <property name="windowTitle">
   <string>qrobot</string>
```

```
</property>
<widget class="QWidget" name="centralWidget"/>
<widget class="QMenuBar" name="menuBar">
 <property name="geometry">
  <rect>
   <x>0</x>
   <y>0</y>
   <width>442</width>
   <height>20</height>
  </rect>
 </property>
</widget>
<widget class="QToolBar" name="mainToolBar">
 <attribute name="toolBarArea">
  <enum>TopToolBarArea</enum>
 </attribute>
 <attribute name="toolBarBreak">
  <bool>false</bool>
 </attribute>
 <addaction name="actionOpen"/>
 <addaction name="actionClose"/>
 <addaction name="actionFoward"/>
 <addaction name="actionBackward"/>
 <addaction name="actionRight"/>
 <addaction name="actionLeft"/>
 <addaction name="actionStop"/>
</widget>
<widget class="QStatusBar" name="statusBar"/>
<action name="actionOpen">
 <property name="text">
  <string>Open</string>
 </property>
 <property name="shortcut">
  <string>Ctrl+O</string>
 </property>
</action>
<action name="actionClose">
 <property name="text">
  <string>Close</string>
 </property>
 <property name="shortcut">
  <string>Ctrl+C</string>
 </property>
</action>
<action name="actionFoward">
 <property name="text">
```

```
      <string>Foward</string>
     </property>
     <property name="shortcut">
      <string>Ctrl+W</string>
     </property>
    </action>
    <action name="actionBackward">
     <property name="text">
      <string>Backward</string>
     </property>
     <property name="shortcut">
      <string>Ctrl+S</string>
     </property>
    </action>
    <action name="actionRight">
     <property name="text">
      <string>Right</string>
     </property>
     <property name="shortcut">
      <string>Ctrl+D</string>
     </property>
    </action>
    <action name="actionLeft">
     <property name="text">
      <string>Left</string>
     </property>
     <property name="shortcut">
      <string>Ctrl+A</string>
     </property>
    </action>
    <action name="actionStop">
     <property name="text">
      <string>Stop</string>
     </property>
     <property name="shortcut">
      <string>Ctrl+X</string>
     </property>
    </action>
   </widget>
   <layoutdefault spacing="6" margin="11"/>
   <resources/>
   <connections/>
  </ui>
```

File main.cpp

```
#include "qrobot.h"
#include <QApplication>

int main(int argc, char *argv[]){
    QApplication a(argc, argv);
    qrobot w;
    w.show();

    return a.exec();
   }
```

Halaman ini memang sengaja dikosongkan

LAMPIRAN C: KODE SUMBER C UNTUK FIRMWARE MOTOR CONTROLLER

Seluruh file ini dapat di akses di URL:
https://github.com/mekatronik-achmadi/tugas_akhir/tree/master/software/rtos

Seluruh kode sumber ini dibangun dengan operating system ubuntu atau debian. Kode sumber ini dibangun berdasarkan pustaka ChibiOS/RT yang merupakan pustaka untuk membangun firmware pada mikrokontroller berbasis ARM-Cortex M-series. Kode sumber yang terlampir disini hanyalah kode sumber yang dibuat dalam perancangan ini dan bukan termasuk kode bawaan dari pustaka ChibiOS/RT. File-file tersebut adalah:

1. halconf.h
2. mcuconf.h
3. robot_gpt.h
4. robot_led.h
5. robot_pal.h
6. robot_shell.h
7. srcconf.h
8. main.c
9. robot_gpt.c
10. robot_led.h
11. robot_pal.c
12. robot_shell.c
13. Makefile

Untuk dapat mengkompilasi kode sumber ini maka dibutuhkan paket berikut:

4. gcc-arm-none-eabi
5. libnewlib-arm-none-eabi
6. libnewlib-dev
7. build-essential

Untuk file kode sumber selain yang terlampir disini maka file tersebut merupakan bagian dari pustaka ChibiOS/RT yang digunakan

tanpa ada perubahan.Untuk mengkompilasinya tinggal memanggil program make.

1. File halconf.h

```
/**
 * @file    templates/halconf.h
 * @brief   HAL configuration header.
 * @details HAL configuration file, this file allows
to enable or disable the
 *          various device drivers from your
application. You may also use
 *          this file in order to override the device
drivers default settings.
 *
 * @addtogroup HAL_CONF
 * @{
 */

#ifndef _HALCONF_H_
#define _HALCONF_H_

#include "mcuconf.h"

/**
 * @brief   Enables the TM subsystem.
 */
#if !defined(HAL_USE_TM) || defined(__DOXYGEN__)
#define HAL_USE_TM                  TRUE
#endif

/**
 * @brief   Enables the PAL subsystem.
 */
#if !defined(HAL_USE_PAL) || defined(__DOXYGEN__)
#define HAL_USE_PAL                 TRUE
#endif

/**
 * @brief   Enables the ADC subsystem.
 */
#if !defined(HAL_USE_ADC) || defined(__DOXYGEN__)
#define HAL_USE_ADC                 FALSE
#endif
```

```
/**
 * @brief   Enables the CAN subsystem.
 */
#if !defined(HAL_USE_CAN) || defined(__DOXYGEN__)
#define HAL_USE_CAN                 FALSE
#endif

/**
 * @brief   Enables the EXT subsystem.
 */
#if !defined(HAL_USE_EXT) || defined(__DOXYGEN__)
#define HAL_USE_EXT                 FALSE
#endif

/**
 * @brief   Enables the GPT subsystem.
 */
#if !defined(HAL_USE_GPT) || defined(__DOXYGEN__)
#define HAL_USE_GPT                 TRUE
#endif

/**
 * @brief   Enables the I2C subsystem.
 */
#if !defined(HAL_USE_I2C) || defined(__DOXYGEN__)
#define HAL_USE_I2C                 FALSE
#endif

/**
 * @brief   Enables the ICU subsystem.
 */
#if !defined(HAL_USE_ICU) || defined(__DOXYGEN__)
#define HAL_USE_ICU                 FALSE
#endif

/**
 * @brief   Enables the MAC subsystem.
 */
#if !defined(HAL_USE_MAC) || defined(__DOXYGEN__)
#define HAL_USE_MAC                 FALSE
#endif

/**
 * @brief   Enables the MMC_SPI subsystem.
 */
#if !defined(HAL_USE_MMC_SPI) || defined(__DOXYGEN__)
```

```
#define HAL_USE_MMC_SPI             FALSE
#endif

/**
 * @brief   Enables the PWM subsystem.
 */
#if !defined(HAL_USE_PWM) || defined(__DOXYGEN__)
#define HAL_USE_PWM                 FALSE
#endif

/**
 * @brief   Enables the RTC subsystem.
 */
#if !defined(HAL_USE_RTC) || defined(__DOXYGEN__)
#define HAL_USE_RTC                 FALSE
#endif

/**
 * @brief   Enables the SDC subsystem.
 */
#if !defined(HAL_USE_SDC) || defined(__DOXYGEN__)
#define HAL_USE_SDC                 FALSE
#endif

/**
 * @brief   Enables the SERIAL subsystem.
 */
#if !defined(HAL_USE_SERIAL) || defined(__DOXYGEN__)
#define HAL_USE_SERIAL              TRUE
#endif

/**
 * @brief   Enables the SERIAL over USB subsystem.
 */
#if !defined(HAL_USE_SERIAL_USB) ||
defined(__DOXYGEN__)
#define HAL_USE_SERIAL_USB          FALSE
#endif

/**
 * @brief   Enables the SPI subsystem.
 */
#if !defined(HAL_USE_SPI) || defined(__DOXYGEN__)
#define HAL_USE_SPI                 FALSE
#endif
```

```
/**
 * @brief   Enables the UART subsystem.
 */
#if !defined(HAL_USE_UART) || defined(__DOXYGEN__)
#define HAL_USE_UART                FALSE
#endif

/**
 * @brief   Enables the USB subsystem.
 */
#if !defined(HAL_USE_USB) || defined(__DOXYGEN__)
#define HAL_USE_USB                 FALSE
#endif

/*===========================================================
========================*/
/* ADC driver related settings.
*/
/*===========================================================
========================*/

/**
 * @brief   Enables synchronous APIs.
 * @note    Disabling this option saves both code and
data space.
 */
#if !defined(ADC_USE_WAIT) || defined(__DOXYGEN__)
#define ADC_USE_WAIT                TRUE
#endif

/**
 * @brief   Enables the @p adcAcquireBus() and @p
adcReleaseBus() APIs.
 * @note    Disabling this option saves both code and
data space.
 */
#if !defined(ADC_USE_MUTUAL_EXCLUSION) ||
defined(__DOXYGEN__)
#define ADC_USE_MUTUAL_EXCLUSION    TRUE
#endif

/*===========================================================
========================*/
/* CAN driver related settings.
*/
/*===========================================================
```

```
========================*/

/**
 * @brief   Sleep mode related APIs inclusion switch.
 */
#if !defined(CAN_USE_SLEEP_MODE) ||
defined(__DOXYGEN__)
#define CAN_USE_SLEEP_MODE          TRUE
#endif

/*=======================================================
========================*/
/* I2C driver related settings.
*/
/*=======================================================
========================*/

/**
 * @brief   Enables the mutual exclusion APIs on the
I2C bus.
 */
#if !defined(I2C_USE_MUTUAL_EXCLUSION) ||
defined(__DOXYGEN__)
#define I2C_USE_MUTUAL_EXCLUSION    TRUE
#endif

/*=======================================================
========================*/
/* MAC driver related settings.
*/
/*=======================================================
========================*/

/**
 * @brief   Enables an event sources for incoming
packets.
 */
#if !defined(MAC_USE_ZERO_COPY) ||
defined(__DOXYGEN__)
#define MAC_USE_ZERO_COPY           FALSE
#endif

/**
 * @brief   Enables an event sources for incoming
packets.
 */
```

```
#if !defined(MAC_USE_EVENTS) || defined(__DOXYGEN__)
#define MAC_USE_EVENTS              TRUE
#endif

/*===========================================================
=========================*/
/* MMC_SPI driver related settings.
*/
/*===========================================================
=========================*/

/**
 * @brief   Delays insertions.
 * @details If enabled this options inserts delays
into the MMC waiting
 *          routines releasing some extra CPU time
for the threads with
 *          lower priority, this may slow down the
driver a bit however.
 *          This option is recommended also if the
SPI driver does not
 *          use a DMA channel and heavily loads the
CPU.
 */
#if !defined(MMC_NICE_WAITING) ||
defined(__DOXYGEN__)
#define MMC_NICE_WAITING            TRUE
#endif

/*===========================================================
=========================*/
/* SDC driver related settings.
*/
/*===========================================================
=========================*/

/**
 * @brief   Number of initialization attempts before
rejecting the card.
 * @note    Attempts are performed at 10mS intervals.
 */
#if !defined(SDC_INIT_RETRY) || defined(__DOXYGEN__)
#define SDC_INIT_RETRY              100
#endif

/**
```

```
 * @brief   Include support for MMC cards.
 * @note    MMC support is not yet implemented so
this option must be kept
 *          at @p FALSE.
 */
#if !defined(SDC_MMC_SUPPORT) || defined(__DOXYGEN__)
#define SDC_MMC_SUPPORT             FALSE
#endif

/**
 * @brief   Delays insertions.
 * @details If enabled this options inserts delays
into the MMC waiting
 *          routines releasing some extra CPU time
for the threads with
 *          lower priority, this may slow down the
driver a bit however.
 */
#if !defined(SDC_NICE_WAITING) ||
defined(__DOXYGEN__)
#define SDC_NICE_WAITING            TRUE
#endif

/*==========================================================
=========================*/
/* SERIAL driver related settings.
*/
/*==========================================================
=========================*/

/**
 * @brief   Default bit rate.
 * @details Configuration parameter, this is the baud
rate selected for the
 *          default configuration.
 */
#if !defined(SERIAL_DEFAULT_BITRATE) ||
defined(__DOXYGEN__)
#define SERIAL_DEFAULT_BITRATE      38400
#endif

/**
 * @brief   Serial buffers size.
 * @details Configuration parameter, you can change
the depth of the queue
 *          buffers depending on the requirements of
```

```
your application.
 * @note    The default is 64 bytes for both the
transmission and receive
 *          buffers.
 */
#if !defined(SERIAL_BUFFERS_SIZE) ||
defined(__DOXYGEN__)
#define SERIAL_BUFFERS_SIZE         32
#endif

/*==========================================================
=========================*/
/* SPI driver related settings.
*/
/*==========================================================
=========================*/

/**
 * @brief   Enables synchronous APIs.
 * @note    Disabling this option saves both code and
data space.
 */
#if !defined(SPI_USE_WAIT) || defined(__DOXYGEN__)
#define SPI_USE_WAIT                TRUE
#endif

/**
 * @brief   Enables the @p spiAcquireBus() and @p
spiReleaseBus() APIs.
 * @note    Disabling this option saves both code and
data space.
 */
#if !defined(SPI_USE_MUTUAL_EXCLUSION) ||
defined(__DOXYGEN__)
#define SPI_USE_MUTUAL_EXCLUSION    TRUE
#endif

#endif /* _HALCONF_H_ */

/** @} */
```

2. File mcuconf.h

```
#define STM32F103_MCUCONF

/*
```

```
 * STM32F103 drivers configuration.
 * The following settings override the default settings
present in
 * the various device driver implementation headers.
 * Note that the settings for each driver only have
effect if the whole
 * driver is enabled in halconf.h.
 *
 * IRQ priorities:
 * 15...0       Lowest...Highest.
 *
 * DMA priorities:
 * 0...3        Lowest...Highest.
 */

/*
 * HAL driver system settings.
 */
#define STM32_NO_INIT                       FALSE
#define STM32_HSI_ENABLED                   TRUE
#define STM32_LSI_ENABLED                   FALSE
#define STM32_HSE_ENABLED                   TRUE
#define STM32_LSE_ENABLED                   FALSE
#define STM32_SW                            STM32_SW_PLL
#define STM32_PLLSRC
STM32_PLLSRC_HSE
#define STM32_PLLXTPRE
STM32_PLLXTPRE_DIV1
#define STM32_PLLMUL_VALUE                  9
#define STM32_HPRE
STM32_HPRE_DIV1
#define STM32_PPRE1
STM32_PPRE1_DIV2
#define STM32_PPRE2
STM32_PPRE2_DIV2
#define STM32_ADCPRE
STM32_ADCPRE_DIV4
#define STM32_USB_CLOCK_REQUIRED            TRUE
#define STM32_USBPRE
STM32_USBPRE_DIV1P5
#define STM32_MCOSEL
STM32_MCOSEL_NOCLOCK
#define STM32_RTCSEL
STM32_RTCSEL_HSEDIV
#define STM32_PVD_ENABLE                    FALSE
#define STM32_PLS
```

```
STM32_PLS_LEV0

/*
 * ADC driver system settings.
 */
#define STM32_ADC_USE_ADC1                  FALSE
#define STM32_ADC_ADC1_DMA_PRIORITY         2
#define STM32_ADC_ADC1_IRQ_PRIORITY         6

/*
 * CAN driver system settings.
 */
#define STM32_CAN_USE_CAN1                  FALSE
#define STM32_CAN_CAN1_IRQ_PRIORITY         11

/*
 * EXT driver system settings.
 */
#define STM32_EXT_EXTI0_IRQ_PRIORITY        6
#define STM32_EXT_EXTI1_IRQ_PRIORITY        6
#define STM32_EXT_EXTI2_IRQ_PRIORITY        6
#define STM32_EXT_EXTI3_IRQ_PRIORITY        6
#define STM32_EXT_EXTI4_IRQ_PRIORITY        6
#define STM32_EXT_EXTI5_9_IRQ_PRIORITY      6
#define STM32_EXT_EXTI10_15_IRQ_PRIORITY    6
#define STM32_EXT_EXTI16_IRQ_PRIORITY       6
#define STM32_EXT_EXTI17_IRQ_PRIORITY       6
#define STM32_EXT_EXTI18_IRQ_PRIORITY       6
#define STM32_EXT_EXTI19_IRQ_PRIORITY       6

/*
 * GPT driver system settings.
 */
#define STM32_GPT_USE_TIM1                  FALSE
#define STM32_GPT_USE_TIM2                  FALSE
#define STM32_GPT_USE_TIM3                  TRUE
#define STM32_GPT_USE_TIM4                  FALSE
#define STM32_GPT_USE_TIM5                  FALSE
#define STM32_GPT_USE_TIM8                  FALSE
#define STM32_GPT_TIM1_IRQ_PRIORITY         7
#define STM32_GPT_TIM2_IRQ_PRIORITY         7
#define STM32_GPT_TIM3_IRQ_PRIORITY         7
#define STM32_GPT_TIM4_IRQ_PRIORITY         7
#define STM32_GPT_TIM5_IRQ_PRIORITY         7
#define STM32_GPT_TIM8_IRQ_PRIORITY         7
```

```
/*
 * I2C driver system settings.
 */
#define STM32_I2C_USE_I2C1                  FALSE
#define STM32_I2C_USE_I2C2                  FALSE
#define STM32_I2C_I2C1_IRQ_PRIORITY         5
#define STM32_I2C_I2C2_IRQ_PRIORITY         5
#define STM32_I2C_I2C1_DMA_PRIORITY         3
#define STM32_I2C_I2C2_DMA_PRIORITY         3
#define STM32_I2C_I2C1_DMA_ERROR_HOOK()     chSysHalt()
#define STM32_I2C_I2C2_DMA_ERROR_HOOK()     chSysHalt()

/*
 * ICU driver system settings.
 */
#define STM32_ICU_USE_TIM1                  FALSE
#define STM32_ICU_USE_TIM2                  FALSE
#define STM32_ICU_USE_TIM3                  FALSE
#define STM32_ICU_USE_TIM4                  FALSE
#define STM32_ICU_USE_TIM5                  FALSE
#define STM32_ICU_USE_TIM8                  FALSE
#define STM32_ICU_TIM1_IRQ_PRIORITY         7
#define STM32_ICU_TIM2_IRQ_PRIORITY         7
#define STM32_ICU_TIM3_IRQ_PRIORITY         7
#define STM32_ICU_TIM4_IRQ_PRIORITY         7
#define STM32_ICU_TIM5_IRQ_PRIORITY         7
#define STM32_ICU_TIM8_IRQ_PRIORITY         7

/*
 * PWM driver system settings.
 */
#define STM32_PWM_USE_ADVANCED              FALSE
#define STM32_PWM_USE_TIM1                  FALSE
#define STM32_PWM_USE_TIM2                  FALSE
#define STM32_PWM_USE_TIM3                  FALSE
#define STM32_PWM_USE_TIM4                  FALSE
#define STM32_PWM_USE_TIM5                  FALSE
#define STM32_PWM_USE_TIM8                  FALSE
#define STM32_PWM_TIM1_IRQ_PRIORITY         7
#define STM32_PWM_TIM2_IRQ_PRIORITY         7
#define STM32_PWM_TIM3_IRQ_PRIORITY         7
#define STM32_PWM_TIM4_IRQ_PRIORITY         7
#define STM32_PWM_TIM5_IRQ_PRIORITY         7
#define STM32_PWM_TIM8_IRQ_PRIORITY         7

/*
```

```
 * RTC driver system settings.
 */
#define STM32_RTC_IRQ_PRIORITY              15

/*
 * SERIAL driver system settings.
 */
#define STM32_SERIAL_USE_USART1             TRUE
#define STM32_SERIAL_USE_USART2             FALSE
#define STM32_SERIAL_USE_USART3             FALSE
#define STM32_SERIAL_USE_UART4              FALSE
#define STM32_SERIAL_USE_UART5              FALSE
#define STM32_SERIAL_USART1_PRIORITY        12
#define STM32_SERIAL_USART2_PRIORITY        12
#define STM32_SERIAL_USART3_PRIORITY        12
#define STM32_SERIAL_UART4_PRIORITY         12
#define STM32_SERIAL_UART5_PRIORITY         12

/*
 * SPI driver system settings.
 */
#define STM32_SPI_USE_SPI1                  FALSE
#define STM32_SPI_USE_SPI2                  FALSE
#define STM32_SPI_USE_SPI3                  FALSE
#define STM32_SPI_SPI1_DMA_PRIORITY         1
#define STM32_SPI_SPI2_DMA_PRIORITY         1
#define STM32_SPI_SPI3_DMA_PRIORITY         1
#define STM32_SPI_SPI1_IRQ_PRIORITY         10
#define STM32_SPI_SPI2_IRQ_PRIORITY         10
#define STM32_SPI_SPI3_IRQ_PRIORITY         10
#define STM32_SPI_DMA_ERROR_HOOK(spip)      chSysHalt()

/*
 * UART driver system settings.
 */
#define STM32_UART_USE_USART1               FALSE
#define STM32_UART_USE_USART2               FALSE
#define STM32_UART_USE_USART3               FALSE
#define STM32_UART_USART1_IRQ_PRIORITY      12
#define STM32_UART_USART2_IRQ_PRIORITY      12
#define STM32_UART_USART3_IRQ_PRIORITY      12
#define STM32_UART_USART1_DMA_PRIORITY      0
#define STM32_UART_USART2_DMA_PRIORITY      0
#define STM32_UART_USART3_DMA_PRIORITY      0
#define STM32_UART_DMA_ERROR_HOOK(uartp)    chSysHalt()
```

```
/*
 * USB driver system settings.
 */
#define STM32_USB_USE_USB1                  FALSE
#define STM32_USB_LOW_POWER_ON_SUSPEND      FALSE
#define STM32_USB_USB1_HP_IRQ_PRIORITY      13
#define STM32_USB_USB1_LP_IRQ_PRIORITY      14
```

3. File robot_gpt.h

```
#ifndef ROBOT_GPT
#define ROBOT_GPT

#define F_GPT 100000

void Gpt_Setup(void);

#endif
```

4. File robot_led.h

```
#ifndef ROBOT_LED
#define ROBOT_LED

#define L1 0
#define L2 1
#define L3 2
#define L4 3
#define L5 4
#define L6 5
#define L7 6
#define L8 7

#define L0 8

#define PORT_LED GPIOA

void Led_Setup(void);

#endif
```

8. File robot_pal.h

```
#ifndef ROBOT_PAL
#define ROBOT_PAL
```

```
#define O1 0
#define O2 1
#define O3 2
#define O4 3
#define O5 4
#define O6 5
#define O7 6
#define O8 7

#define O0 8

#define PORT_PAL GPIOC

#define LEFT_A O1
#define LEFT_B O2
#define LEFT_I L1

#define RIGHT_A O3
#define RIGHT_B O4
#define RIGHT_I L2

void Pal_Setup(void);

void Left_fow(void);
void Left_back(void);

void Right_fow(void);
void Right_back(void);

void Pal_clear(void);

#endif
```

9. File robot_shell.h

```
#ifndef ROBOT_SHELL
#define ROBOT_SHELL

#define SHELL_WA_SIZE   THD_WA_SIZE(2048)
#define TEST_WA_SIZE    THD_WA_SIZE(256)

#define STOP        0
#define FOWARD      1
#define BACKWARD    2
#define RIGHT       3
```

```
    #define LEFT        4

    void Serial_Setup(void);
    void Shell_Setup(void);

    #endif
```

10. File srcconf.h

```
#ifndef SRC_CONF
#define SRC_CONF

#include <stdio.h>
#include <string.h>
#include <stdint.h>
#include <stdarg.h>
#include <stdlib.h>
#include <math.h>

#include "ch.h"
#include "hal.h"

#include "memstreams.h"
#include "shell.h"
#include "chprintf.h"
#include "evtimer.h"

#include "robot_led.h"
#include "robot_shell.h"
#include "robot_pal.h"
#include "robot_gpt.h"

#define assert_param(expr) ((void)0)

#endif
```

11. File main.c

```
#include "srcconf.h"

/*
 * Application entry point.
 */
int main(void) {
  /*
   * System initializations.
```

```
   * - HAL initialization, this also initializes the
configured device drivers
   *   and performs the board-specific initializations.
   * - Kernel initialization, the main() function
becomes a thread and the
   *   RTOS is active.
   */
  halInit();
  chSysInit();

  /*
   * Project defined HAL Setup
   */
   Gpt_Setup();
   Pal_Setup();
   Led_Setup();
   Serial_Setup();

  while (TRUE) {
    Shell_Setup(); /* create shell */
    chThdSleepMilliseconds(500);
  };
}
```

12. File robot_gpt.c

```
#include "srcconf.h"

uint16_t loop;
uint16_t value=10;
extern uint16_t dir;

static void gpt3cb(GPTDriver *gptp) {
  (void)gptp;

  loop++;

  if(loop==value){
    Pal_clear();
  }

  if(loop==100){

    loop=0;

    if(dir==FOWARD){
```

```
        Right_fow();
        Left_fow();
        palClearPad(PORT_LED,RIGHT_I);
        palClearPad(PORT_LED,LEFT_I);
      }
      else if(dir==BACKWARD){
        Right_back();
        Left_back();
        palClearPad(PORT_LED,RIGHT_I);
        palClearPad(PORT_LED,LEFT_I);
      }
      else if(dir==LEFT){
        Right_back();
        Left_fow();
        palClearPad(PORT_LED,RIGHT_I);
        palSetPad(PORT_LED,LEFT_I);
      }
      else if(dir==RIGHT){
        Right_fow();
        Left_back();
        palSetPad(PORT_LED,RIGHT_I);
        palClearPad(PORT_LED,LEFT_I);
      }
      else if(dir==STOP){
        Pal_clear();
        palSetPad(PORT_LED,RIGHT_I);
        palSetPad(PORT_LED,LEFT_I);
      }

    }

    chSysLockFromIsr();
    gptStartOneShotI(&GPTD3, 10);
    chSysUnlockFromIsr();
  }

  static const GPTConfig gpt3cfg = {
    F_GPT,    /* 10kHz timer clock.*/
    gpt3cb,   /* Timer callback.*/
    0
  };

  void Gpt_Setup(void){
    gptStart(&GPTD3, &gpt3cfg);
    gptStartOneShot(&GPTD3, 10);
  }
```

13. File robot_led.c

```
#include "srcconf.h"

static WORKING_AREA(wa_ledThread, 128);
static msg_t ledThread(void *arg) {
  (void)arg;

  while(TRUE){
      chThdSleepMilliseconds(500);
      palTogglePad(PORT_LED,L0);
  }

  return 0;
}

static WORKING_AREA(wa_ledTestThread, 128);
static msg_t ledTestThread(void *arg) {
  (void)arg;

  while(TRUE){
      chThdSleepMilliseconds(1000);
      palSetPad(PORT_LED,L1);
      palSetPad(PORT_LED,L2);
      palSetPad(PORT_LED,L3);
      palSetPad(PORT_LED,L4);
      palSetPad(PORT_LED,L5);
      palSetPad(PORT_LED,L6);
      palSetPad(PORT_LED,L7);
      palSetPad(PORT_LED,L8);

      chThdSleepMilliseconds(1000);
      palClearPad(PORT_LED,L1);
      palClearPad(PORT_LED,L2);
      palClearPad(PORT_LED,L3);
      palClearPad(PORT_LED,L4);
      palClearPad(PORT_LED,L5);
      palClearPad(PORT_LED,L6);
      palClearPad(PORT_LED,L7);
      palClearPad(PORT_LED,L8);
  }

  return 0;
}

void Led_Setup(void){
```

```
  palSetPadMode(PORT_LED,L0,PAL_MODE_OUTPUT_PUSHPULL);
  palSetPadMode(PORT_LED,L1,PAL_MODE_OUTPUT_PUSHPULL);
  palSetPadMode(PORT_LED,L2,PAL_MODE_OUTPUT_PUSHPULL);
  palSetPadMode(PORT_LED,L3,PAL_MODE_OUTPUT_PUSHPULL);
  palSetPadMode(PORT_LED,L4,PAL_MODE_OUTPUT_PUSHPULL);
  palSetPadMode(PORT_LED,L5,PAL_MODE_OUTPUT_PUSHPULL);
  palSetPadMode(PORT_LED,L6,PAL_MODE_OUTPUT_PUSHPULL);
  palSetPadMode(PORT_LED,L7,PAL_MODE_OUTPUT_PUSHPULL);
  palSetPadMode(PORT_LED,L8,PAL_MODE_OUTPUT_PUSHPULL);

  palSetPad(PORT_LED,L0);
  palSetPad(PORT_LED,L1);
  palSetPad(PORT_LED,L2);
  palSetPad(PORT_LED,L3);
  palSetPad(PORT_LED,L4);
  palSetPad(PORT_LED,L5);
  palSetPad(PORT_LED,L6);
  palSetPad(PORT_LED,L7);
  palSetPad(PORT_LED,L8);

  chThdCreateStatic(wa_ledThread, sizeof(wa_ledThread),
NORMALPRIO, ledThread, NULL);
  //chThdCreateStatic(wa_ledTestThread,
sizeof(wa_ledTestThread), NORMALPRIO, ledTestThread,
NULL);
}
```

14. File robot_pal.c

```
#include "srcconf.h"

void Right_fow(void){
    palSetPad(PORT_PAL,RIGHT_B);
    palClearPad(PORT_PAL,RIGHT_A);

    palClearPad(PORT_LED,RIGHT_I);
}

void Right_back(void){
    palSetPad(PORT_PAL,RIGHT_A);
    palClearPad(PORT_PAL,RIGHT_B);

    palSetPad(PORT_LED,RIGHT_I);
}

void Left_fow(void){
    palSetPad(PORT_PAL,LEFT_B);
```

```
    palClearPad(PORT_PAL,LEFT_A);

    palClearPad(PORT_LED,LEFT_I);
}

void Left_back(void){
    palSetPad(PORT_PAL,LEFT_A);
    palClearPad(PORT_PAL,LEFT_B);

    palSetPad(PORT_LED,LEFT_I);
}

void Pal_clear(void){
    palClearPad(PORT_PAL,LEFT_A);
    palClearPad(PORT_PAL,LEFT_B);
    palClearPad(PORT_PAL,RIGHT_A);
    palClearPad(PORT_PAL,RIGHT_B);
}

static WORKING_AREA(wa_palTestThread, 128);
static msg_t palTestThread(void *arg) {
  (void)arg;

  while(TRUE){
      Left_back();
      Right_fow();
      chThdSleepMilliseconds(1000);

      Left_fow();
      Right_back();
      chThdSleepMilliseconds(1000);
  }

  return 0;
}

void Pal_Setup(void){

palSetPadMode(PORT_PAL,O1,PAL_MODE_OUTPUT_PUSHPULL);

palSetPadMode(PORT_PAL,O2,PAL_MODE_OUTPUT_PUSHPULL);

palSetPadMode(PORT_PAL,O3,PAL_MODE_OUTPUT_PUSHPULL);

palSetPadMode(PORT_PAL,O4,PAL_MODE_OUTPUT_PUSHPULL);
```

```
    palSetPadMode(PORT_PAL,O5,PAL_MODE_OUTPUT_PUSHPULL);

    palSetPadMode(PORT_PAL,O6,PAL_MODE_OUTPUT_PUSHPULL);

    palSetPadMode(PORT_PAL,O7,PAL_MODE_OUTPUT_PUSHPULL);

    palSetPadMode(PORT_PAL,O8,PAL_MODE_OUTPUT_PUSHPULL);

      //chThdCreateStatic(wa_palTestThread,
    sizeof(wa_palTestThread), NORMALPRIO, palTestThread,
    NULL);
    }
```

15. File robot_shell.c

```
#include "srcconf.h"

uint16_t dir;

Thread *shelltp = NULL;

static void cmd_test(BaseSequentialStream *chp, int
argc, char *argv[]) {
  (void)argv;

  if(argc>0){
    chprintf(chp,"bad commands\n\r");
    return;
  }

  chprintf(chp,"serial ok !!\n\r");
  return;
}

static void cmd_foward(BaseSequentialStream *chp, int
argc, char *argv[]) {
  (void)argv;

  if(argc>0){
    chprintf(chp,"bad commands\n\r");
    return;
  }
  dir=FOWARD;
  chThdSleepMilliseconds(250);
  dir=STOP;
  return;
```

```
}

static void cmd_backward(BaseSequentialStream *chp, int
argc, char *argv[]) {
  (void)argv;

  if(argc>0){
    chprintf(chp,"bad commands\n\r");
    return;
  }
  dir=BACKWARD;
  chThdSleepMilliseconds(250);
  dir=STOP;
  return;
}

static void cmd_right(BaseSequentialStream *chp, int
argc, char *argv[]) {
  (void)argv;

  if(argc>0){
    chprintf(chp,"bad commands\n\r");
    return;
  }
  dir=RIGHT;
  chThdSleepMilliseconds(50);
  dir=STOP;
  return;
}

static void cmd_left(BaseSequentialStream *chp, int
argc, char *argv[]) {
  (void)argv;

  if(argc>0){
    chprintf(chp,"bad commands\n\r");
    return;
  }
  dir=LEFT;
  chThdSleepMilliseconds(50);
  dir=STOP;
  return;
}

static void cmd_stop(BaseSequentialStream *chp, int
argc, char *argv[]) {
```

```
  (void)argv;

  if(argc>0){
    chprintf(chp,"bad commands\n\r");
    return;
  }
  dir=STOP;
  return;
}

static void cmd_booted(BaseSequentialStream *chp, int
argc, char *argv[]) {
  (void)argv;

  if(argc>0){
    chprintf(chp,"bad commands\n\r");
    return;
  }
  palTogglePad(PORT_LED,L3);
  return;
}

static const ShellCommand commands[] = {
  {"test",cmd_test},
  {"foward",cmd_foward},
  {"backward",cmd_backward},
  {"right",cmd_right},
  {"left",cmd_left},
  {"stop",cmd_stop},
  {"booted",cmd_booted},
  {NULL, NULL}
};

static const ShellConfig shell_cfg1 = {
  (BaseSequentialStream *)&SD1,
  commands
};

void Serial_Setup(void){
  palSetPadMode(GPIOA,9,16);
  palSetPadMode(GPIOA,10,2);
  sdStart(&SD1,NULL);
  shellInit();
}

void Shell_Setup(void){
```

```
    if (!shelltp){
      shelltp = shellCreate(&shell_cfg1, SHELL_WA_SIZE,
NORMALPRIO);} /* create shell tread */
    else if (chThdTerminated(shelltp)) {
      chThdRelease(shelltp);    /* Recovers memory of
the previous shell.   */
      shelltp = NULL;           /* Triggers spawning of
a new shell.        */
    }
}
```

16. File Makefile

```
ifeq ($(USE_OPT),)
  USE_OPT = -O2 -ggdb -fomit-frame-pointer -falign-
functions=16
endif

# C specific options here (added to USE_OPT).
ifeq ($(USE_COPT),)
  USE_COPT =
endif

# C++ specific options here (added to USE_OPT).
ifeq ($(USE_CPPOPT),)
  USE_CPPOPT = -fno-rtti
endif

# Enable this if you want the linker to remove unused
code and data
ifeq ($(USE_LINK_GC),)
  USE_LINK_GC = yes
endif

# If enabled, this option allows to compile the
application in THUMB mode.
ifeq ($(USE_THUMB),)
  USE_THUMB = yes
endif

# Enable this if you want to see the full log while
compiling.
ifeq ($(USE_VERBOSE_COMPILE),)
  USE_VERBOSE_COMPILE = no
endif
```

```
#
# Build global options
##############################################################
##########################

##############################################################
##########################
# Architecture or project specific options
#

# Enable this if you really want to use the STM
FWLib.
ifeq ($(USE_FWLIB),)
  USE_FWLIB = no
endif

#
# Architecture or project specific options
##############################################################
##########################

##############################################################
##########################
# Project, sources and paths
#

# Define project name here
PROJECT = robot

# Imported source files and paths
CHIBIOS = ..
include $(CHIBIOS)/boards/STM32/board.mk
include
$(CHIBIOS)/os/hal/platforms/STM32F1xx/platform.mk
include $(CHIBIOS)/os/hal/hal.mk
include
$(CHIBIOS)/os/ports/GCC/ARMCMx/STM32F1xx/port.mk
include $(CHIBIOS)/os/kernel/kernel.mk

# Define linker script file here
LDSCRIPT= $(PORTLD)/STM32F103xB.ld

# C sources that can be compiled in ARM or THUMB mode
depending on the global
# setting.
CSRC = $(PORTSRC) \
```

```
       $(KERNSRC) \
       $(TESTSRC) \
       $(HALSRC) \
       $(PLATFORMSRC) \
       $(BOARDSRC) \
       $(CHIBIOS)/os/various/syscalls.c \
       $(CHIBIOS)/os/various/evtimer.c \
       $(CHIBIOS)/os/various/shell.c \
       $(CHIBIOS)/os/various/chprintf.c \
       $(CHIBIOS)/os/various/memstreams.c \
       main.c robot_led.c robot_pal.c robot_gpt.c
robot_shell.c

# C++ sources that can be compiled in ARM or THUMB
mode depending on the global
# setting.
CPPSRC =

# C sources to be compiled in ARM mode regardless of
the global setting.
# NOTE: Mixing ARM and THUMB mode enables the -
mthumb-interwork compiler
#       option that results in lower performance and
larger code size.
ACSRC =

# C++ sources to be compiled in ARM mode regardless
of the global setting.
# NOTE: Mixing ARM and THUMB mode enables the -
mthumb-interwork compiler
#       option that results in lower performance and
larger code size.
ACPPSRC =

# C sources to be compiled in THUMB mode regardless
of the global setting.
# NOTE: Mixing ARM and THUMB mode enables the -
mthumb-interwork compiler
#       option that results in lower performance and
larger code size.
TCSRC =

# C sources to be compiled in THUMB mode regardless
of the global setting.
# NOTE: Mixing ARM and THUMB mode enables the -
mthumb-interwork compiler
```

```
#       option that results in lower performance and
larger code size.
TCPPSRC =

# List ASM source files here
ASMSRC = $(PORTASM)

INCDIR = $(PORTINC) $(KERNINC) $(TESTINC) \
         $(HALINC) $(PLATFORMINC) $(BOARDINC) \
         $(CHIBIOS)/os/various
#
# Project, sources and paths
##############################################################
##########################

##############################################################
##########################
# Compiler settings
#

MCU  = cortex-m3

#TRGT = arm-elf-
TRGT = arm-none-eabi-
CC   = $(TRGT)gcc
CPPC = $(TRGT)g++
# Enable loading with g++ only if you need C++
runtime support.
# NOTE: You can use C++ even without C++ support if
you are careful. C++
#       runtime support makes code size explode.
LD   = $(TRGT)gcc
#LD   = $(TRGT)g++
CP   = $(TRGT)objcopy
AS   = $(TRGT)gcc -x assembler-with-cpp
OD   = $(TRGT)objdump
HEX  = $(CP) -O ihex
BIN  = $(CP) -O binary

# ARM-specific options here
AOPT =

# THUMB-specific options here
TOPT = -mthumb -DTHUMB

# Define C warning options here
```

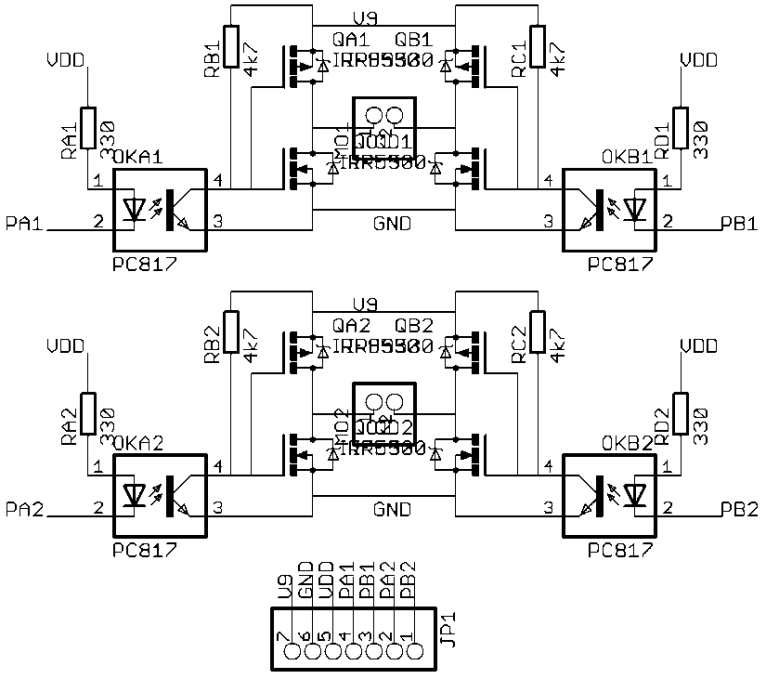
```
CWARN = -Wall -Wextra -Wstrict-prototypes

# Define C++ warning options here
CPPWARN = -Wall -Wextra

#
# Compiler settings
#############################################################
#########################

#############################################################
#########################
# Start of default section
#

# List all default C defines here, like -D_DEBUG=1
DDEFS =

# List all default ASM defines here, like -D_DEBUG=1
DADEFS =

# List all default directories to look for include
files here
DINCDIR =

# List the default directory to look for the
libraries here
DLIBDIR =

# List all default libraries here
DLIBS =

#
# End of default section
#############################################################
#########################

#############################################################
#########################
# Start of user section
#

# List all user C define here, like -D_DEBUG=1
UDEFS =

# Define ASM defines here
```

```
UADEFS =

# List all user directories here
UINCDIR =

# List the user directory to look for the libraries
here
ULIBDIR =

# List all user libraries here
ULIBS =

#
# End of user defines
##############################################################
#########################

ifeq ($(USE_FWLIB),yes)
  include $(CHIBIOS)/ext/stm32lib/stm32lib.mk
  CSRC += $(STM32SRC)
  INCDIR += $(STM32INC)
  USE_OPT += -DUSE_STDPERIPH_DRIVER
endif

include $(CHIBIOS)/os/ports/GCC/ARMCMx/rules.mk
```

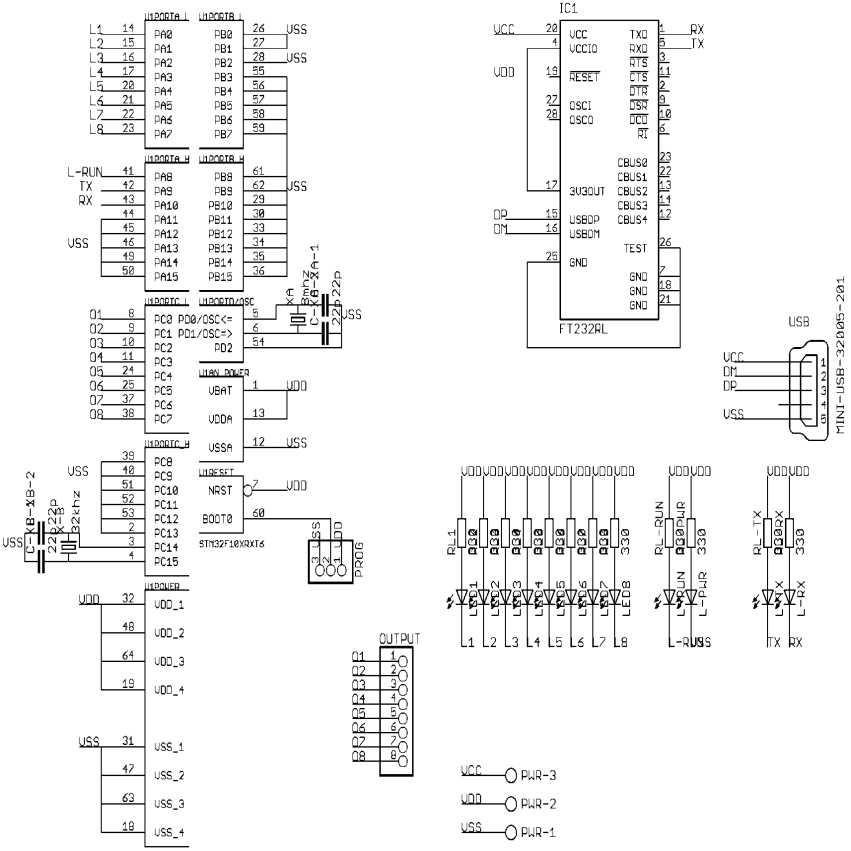
LAMPIRAN D: SKEMA MOTOR DRIVER DAN MOTOR CONTROLLER

Seluruh file ini dapat di akses di URL:
https://github.com/mekatronik-achmadi/tugas_akhir/tree/master/software/driver
Desain ini dirancang menggunakan software EAGLE 6.5 di operating system Ubuntu.

1. Skema Motor Driver



2. Skema Motor Controller



LAMPIRAN E: RINGKASAN DATASHEET KOMPONEN UTAMA

1. STM32F103RBT6

life.augmented

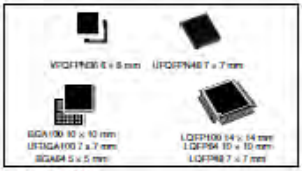
STM32F103x8
STM32F103xB

Medium-density performance line ARM-based 32-bit MCU with 64 or 128 KB Flash, USB, CAN, 7 timers, 2 ADCs, 9 com. interfaces

Datasheet - production data

Features

- ARM 32-bit Cortex™-M3 CPU Core
  - 72 MHz maximum frequency, 1.25 DMIPS/MHz (Dhrystone 2.1) performance at 0 wait state memory access
  - Single-cycle multiplication and hardware division
- Memories
  - 64 or 128 Kbytes of Flash memory
  - 20 Kbytes of SRAM
- Clock, reset and supply management
  - 2.0 to 3.6 V application supply and I/Os
  - POR, PDR, and programmable voltage detector (PVD)
  - 4-to-16 MHz crystal oscillator
  - Internal 8 MHz factory-trimmed RC
  - Internal 40 kHz RC
  - PLL for CPU clock
  - 32 kHz oscillator for RTC with calibration
- Low power
  - Sleep, Stop and Standby modes
  - V_{BAT} supply for RTC and backup registers
- 2 x 12-bit, 1 µs A/D converters (up to 16 channels)
  - Conversion range: 0 to 3.6 V
  - Dual-sample and hold capability
  - Temperature sensor
- DMA
  - 7-channel DMA controller
  - Peripherals supported: timers, ADC, SPIs, I²Cs and USARTs
- Up to 80 fast I/O ports
  - 26/37/51/80 I/Os, all mappable on 16 external interrupt vectors and almost all 5 V-tolerant



- Debug mode
  - Serial wire debug (SWD) & JTAG interfaces
- 7 timers
  - Three 16-bit timers, each with up to 4 IC/OC/PWM or pulse counter and quadrature (incremental) encoder input
  - 16-bit, motor control PWM timer with dead-time generation and emergency stop
  - 2 watchdog timers (Independent and Window)
  - SysTick timer 24-bit downcounter
- Up to 9 communication interfaces
  - Up to 2 x I²C interfaces (SMBus/PMBus)
  - Up to 3 USARTs (ISO 7816 interface, LIN, IrDA capability, modem control)
  - Up to 2 SPIs (18 Mbit/s)
  - CAN interface (2.0B Active)
  - USB 2.0 full-speed interface
- CRC calculation unit, 96-bit unique ID
- Packages are ECOPACK®

Table 1. Device summary

Reference	Part number
STM32F103x8	STM32F103C8, STM32F103R8, STM32F103V8, STM32F103T8
STM32F103xB	STM32F103RB, STM32F103VB, STM32F103CB, STM32F103TB

2. FT232RL

FTDI Chip

Document No.: FT_000053
FT232R USB UART IC Datasheet Version 2.10
Clearance No.: FTDI# 38

Future Technology Devices International Ltd.

FT232R USB UART IC

The FT232R is a USB to serial UART interface with the following advanced features:

- Single chip USB to asynchronous serial data transfer interface.
- Entire USB protocol handled on the chip. No USB specific firmware programming required.
- Fully integrated 1024 bit EEPROM storing device descriptors and CBUS I/O configuration.
- Fully integrated USB termination resistors.
- Fully integrated clock generation with no external crystal required plus optional clock output selection enabling a glue-less interface to external MCU or FPGA.
- Data transfer rates from 300 baud to 3 Mbaud (RS422, RS485, RS232) at TTL levels.
- 128 byte receive buffer and 256 byte transmit buffer utilising buffer smoothing technology to allow for high data throughput.
- FTDI's royalty-free Virtual Com Port (VCP) and Direct (D2XX) drivers eliminate the requirement for USB driver development in most cases.
- Unique USB FTDIChip-ID™ feature.
- Configurable CBUS I/O pins.
- Transmit and receive LED drive signals.
- UART interface support for 7 or 8 data bits, 1 or 2 stop bits and odd / even / mark / space / no parity
- FIFO receive and transmit buffers for high data throughput.
- Synchronous and asynchronous bit bang interface options with RD# and WR# strobes.
- Device supplied pre-programmed with unique USB serial number.
- Supports bus powered, self powered and high-power bus powered USB configurations.
- Integrated +3.3V level converter for USB I/O.
- Integrated level converter on UART and CBUS for interfacing to between +1.8V and +5V logic.
- True 5V/3.3V/2.8V/1.8V CMOS drive output and TTL input.
- Configurable I/O pin output drive strength.
- Integrated power-on-reset circuit.
- Fully integrated AVCC supply filtering - no external filtering required.
- UART signal inversion option.
- +3.3V (using external oscillator) to +5.25V (Internal oscillator) Single Supply Operation.
- Low operating and USB suspend current.
- Low USB bandwidth consumption.
- UHCI/OHCI/EHCI host controller compatible.
- USB 2.0 Full Speed compatible.
- -40°C to 85°C extended operating temperature range.
- Available in compact Pb-free 28 Pin SSOP and QFN-32 packages (both RoHS compliant).

3. RaspberryPi

Raspberry Pi

MODEL B

Product Name	**Raspberry Pi Model B**
Product Description	The Raspberry Pi is a small, powerful and lightweight ARM based computer which can do many of the things a desktop PC can do. The powerful graphics capabilities and HDMI video output make it ideal for multimedia applications such as media centres and narrowcasting solutions. The Raspberry Pi is based on a Broadcom BCM2835 chip. It does not feature a built-in hard disk or solid state drive. Instead relying on an SD card for booting and long-term storage.
RS Part Number	**756-8308**
Specifications	
Chip	Broadcom BCM2835 SoC (a)
Core architecture	ARM11
CPU	700 MHz Low Power ARM1176JZFS Applications Processor
GPU	Dual Core VideoCore IV® Multimedia Co-Processor Provides Open GL ES 2.0, hardware-accelerated OpenVG, and 1080p30 H.264 high-profile decode Capable of 1Gpixel/s, 1.5Gtexel/s or 24GFLOPs with texture filtering and DMA infrastructure
Memory	512MB SDRAM
Operating System	Boots from SD card, running a version of the Linux operating system
Dimensions	85.6 x 53.98 x 17mm
Power	Micro USB socket 5V, 1.2A (i)
Connectors:	
Ethernet	10/100 Base-T Ethernet socket (b)
Video Output	HDMI (rev 1.3 & 1.4) (c); Composite RCA (PAL and NTSC) (d)
Audio Output	3.5mm jack (e), HDMI
USB 2.0	Dual USB Connector (f)
GPIO Connector	26-pin 2.54 mm (100 mil) expansion header: 2x13 strip. Providing 8 GPIO pins plus access to PC, SPI and UART as well as +3.3 V, +5 V and GND supply lines (g)
Camera Connector	15-pin MIPI Camera Serial Interface (CSI-2) (h)
JTAG	Not populated (i)
Display Connector	Display Serial Interface (DSI) 15 way flat flex cable connector with two data lanes and a clock lane (j)
Memory Card Slot	SDIO (k)

www.rs-components.com/raspberrypi

Accessories

▲ Camera Module
775-7731

▲ International power supply
765-3311

▲ 8GB SD card pre-programmed with NOOBS - **770-6770**

▲ Expansion board
772-2974

▲ WiFi dongle
760-3621

▲ 10400mAh Li-Ion battery pack
775-7517

▲ Raspberry Pi user guide
768-6686

RS

4. IRF530

IRF530
IRF530FI

N - CHANNEL ENHANCEMENT MODE POWER MOS TRANSISTOR

TYPE	V_{DSS}	$R_{DS(on)}$	I_D
IRF530	100 V	< 0.16 Ω	16 A
IRF530FI	100 V	< 0.16 Ω	11 A

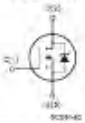
- TYPICAL $R_{DS(on)}$ = 0.12 Ω
- AVALANCHE RUGGED TECHNOLOGY
- 100% AVALANCHE TESTED
- REPETITIVE AVALANCHE DATA AT 100°C
- LOW GATE CHARGE
- HIGH CURRENT CAPABILITY
- 175°C OPERATING TEMPERATURE
- APPLICATION ORIENTED CHARACTERIZATION

TO-220 ISOWATT220

APPLICATIONS

- HIGH CURRENT, HIGH SPEED SWITCHING
- SOLENOID AND RELAY DRIVERS
- DC-DC & DC-AC CONVERTER
- AUTOMOTIVE ENVIRONMENT (INJECTION, ABS, AIR-BAG, LAMP DRIVERS Etc.)

INTERNAL SCHEMATIC DIAGRAM



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value		Unit
		IRF530	IRF530FI	
V_{DS}	Drain-source Voltage (V_{GS} = 0)	100		V
V_{DGR}	Drain- gate Voltage (R_{GS} = 20 kΩ)	100		V
V_{GS}	Gate-source Voltage	± 20		V
I_D	Drain Current (continuous) at T_c = 25 °C	16	11	A
I_D	Drain Current (continuous) at T_c = 100 °C	11	7.8	A
I_{DM}(•)	Drain Current (pulsed)	64	64	A
P_{tot}	Total Dissipation at T_c = 25 °C	90	40	W
	Derating Factor	0.6	0.27	W/°C
V_{ISO}	Insulation Withstand Voltage (DC)	-	2000	V
T_{stg}	Storage Temperature	-65 to 175		°C
T_j	Max. Operating Junction Temperature	175		°C

(•) Pulse width limited by safe operating area (1) I_{SD} ≤16 A, di/dt ≤200 A/µs, V_{DD} ≤ $V_{(BR)DSS}$, T_j ≤ T_{JMAX}

5. IRF9530

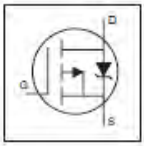
International
IOR Rectifier

PD-91482C

IRF9530N

HEXFET® Power MOSFET

- Advanced Process Technology
- Dynamic dv/dt Rating
- 175°C Operating Temperature
- Fast Switching
- P-Channel
- Fully Avalanche Rated



V_{DSS} = -100V

$R_{DS(on)}$ = 0.20Ω

I_D = -14A

Description

Fifth Generation HEXFETs from International Rectifier utilize advanced processing techniques to achieve extremely low on-resistance per silicon area. This benefit, combined with the fast switching speed and ruggedized device design that HEXFET Power MOSFETs are well known for, provides the designer with an extremely efficient and reliable device for use in a wide variety of applications.

The TO-220 package is universally preferred for all commercial-industrial applications at power dissipation levels to approximately 50 watts. The low thermal resistance and low package cost of the TO-220 contribute to its wide acceptance throughout the industry.

TO-220AB

Absolute Maximum Ratings

	Parameter	Max.	Units
I_D @ T_C = 25°C	Continuous Drain Current, V_{GS} @ -10V	-14	A
I_D @ T_C = 100°C	Continuous Drain Current, V_{GS} @ -10V	-10	
I_{DM}	Pulsed Drain Current ①	-56	
P_D @T_C = 25°C	Power Dissipation	79	W
	Linear Derating Factor	0.53	W/°C
V_{GS}	Gate-to-Source Voltage	± 20	V
E_{AS}	Single Pulse Avalanche Energy②	250	mJ
I_{AR}	Avalanche Current①	-8.4	A
E_{AR}	Repetitive Avalanche Energy①	7.9	mJ
dv/dt	Peak Diode Recovery dv/dt ③	-5.0	V/ns
T_J T_{STG}	Operating Junction and Storage Temperature Range	-55 to + 175	°C
	Soldering Temperature, for 10 seconds	300 (1.6mm from case)	
	Mounting torque, 6-32 or M3 screw	10 lbf•in (1.1N•m)	

Thermal Resistance

	Parameter	Typ.	Max.	Units
$R_{\theta JC}$	Junction-to-Case	—	1.9	°C/W
$R_{\theta CS}$	Case-to-Sink, Flat, Greased Surface	0.50	—	
$R_{\theta JA}$	Junction-to-Ambient	—	62	

5/13/98

6. PC817

PC817 Series

High Density Mounting Type Photocoupler

● Lead forming type (I type) and taping reel type (P type) are also available. (PC817I/PC817P)
●● TÜV (VDE0884) approved type is also available as an option.

■ Features

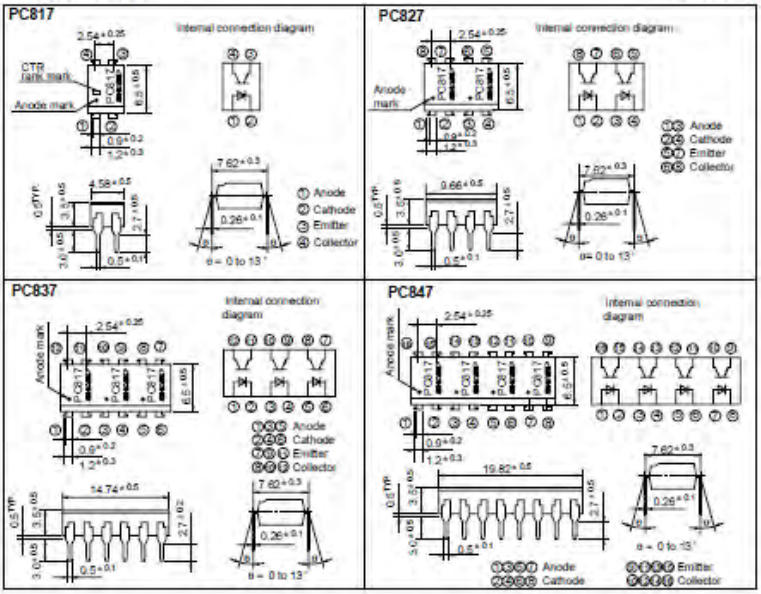
1. Current transfer ratio
   (CTR: MIN. 50% at I_F = 5mA, V_{CE}=5V)
2. High isolation voltage between input and output (V_{iso}: 5 000V$_{rms}$)
3. Compact dual-in-line package
   PC817: 1-channel type
   PC827: 2-channel type
   PC837: 3-channel type
   PC847: 4-channel type
4. Recognized by UL, file No. E64380

■ Applications

1. Computer terminals
2. System appliances, measuring instruments
3. Registers, copiers, automatic vending machines
4. Electric home appliances, such as fan heaters, etc.
5. Signal transmission between circuits of different potentials and impedances

■ Outline Dimensions

(Unit : mm)



DAFTAR PUSTAKA

Aldebaran-Robotic. (2012). *Nao Technical Datasheet.* France: Aldebaran Press.

Andor-Team. (2012). *Digital Camera Fundamentals.* USA: Andor Publisher.

Browning, B. (2013). *Real Time, Adaptive Color-based Robot Vision.* USA: Carnigie Mellon University.

Chao, F. (2014). *A developmental approach to robotic pointing via humanrobot interaction.* China: Xiamen University.

Honda-Team. (2007). *Asimo Techinal Information.* Japan: Honda Press.

OpenCV-Team. (2014). *The OpenCV Reference.* USA: OpenCV.org.

Philips, D. (2000). *Image Proccessing in C.* USA: R&D Publisher.

Robotis. (2012). *Darwin-Op Manual.* Korea: Robotis Inc.

Ude, A. (2010). *Robot Vision.* Croatia: In-Tech.

Upton, E. (2012). *RaspberryPi Guide.* UK: RaspberryPi.org.

Zhou, H. (2010). *Digital Image Proccessing.* USA: BookBon.

Halaman ini memang sengaja dikosongkan

BIOGRAFI PENULIS

Achmadi - lahir di Surabaya, 15 November 1990 jam 5 pagi pada suasana turun hujan. Penulis merupakan anak pertama dari Suwarnu dan Mukarromah. Penulis memulai pendidikan di TK Dharma Wanita Bluto di tahun 1995, dilanjutkan ke SDN Bluto I Kota Sumenep pada tahun 1997 hingga 2003. Kemudian melanjutkan pendidikan di SMPN 1 Sumenep dan lulus pada tahun 2006. Pada tahun yang sama penulis melanjutkan pendidikan di SMAN 3 Pamekasan hingga lulus pada tahun 2009. Selepas lulus dari sekolah menengah atas, penulis melanjutkan pendidikan di Universitas Negeri Surabaya jurusan Fisika Non-Pendidikan hingga tahun 2010. Panggilan jiwa untuk terjun dalam dunia teknik membawa penulis ke Teknik Fisika ITS dari tahun 2010 hingga 2015. Selama menjadi mahasiswa teknik, penulis sempat menjadi anggota Tim Mekatronik ITS untuk divisi Kompetisi Robot Humanoid Soccer sebagai Programmer. Kemudian penulis juga sempat menjadi anggota tim Sapu Angin untuk kompetisi Japan Formula Student sebagai Electric Instrument Engineer. Karya yang pernah dikembangkan penulis antara lain adalah Engine Control Unit untuk mesin injeksi Sinjay buatan Teknik Mesin ITS dan Digital PowerMeter untuk kompetisi Mobil Listrik IEMC 2013. Saat ini penulis aktif dalam pengembangan sistem tertanam dan perangkat lunak antar-muka berbasis opensources. Penulis dapat dihubungi di facebook www.facebook.com/arramadhandevelopment dan email mekatronik.achmadi@gmail.com. Untuk menggunakan karya penulis yang bersifat opensourse dapat diakses di alamat github.com/mekatronik-achmadi

Halaman ini memang sengaja dikosongkan